NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

THE AS FINANCIAL REPORTING SYSTEM: SOME EXPERIENCE ON PROTOTYPING AND USER INTERACTION

by

Ronald L. Booker

March 1986

Thesis Advisor: Tung Bui
Co-advisor: Norman F. Schneidewind

Approved for public release; distribution is unlimited.

ECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

a. REPORT SECURITY CLASSIFICATION	1b. RESTRICTIVE MARKINGS
a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT Approved for public release; distribution is unlimited
b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)

a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School	6b. OFFICE SYMBOL (*If applicable*) Code 54	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
c. ADDRESS (*City, State, and ZIP Code*) Monterey, California 93943-5000		7b. ADDRESS (*City, State, and ZIP Code*) Monterey, California 93943-5000
a. NAME OF FUNDING / SPONSORING ORGANIZATION	8b. OFFICE SYMBOL (*If applicable*)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
c. ADDRESS (*City, State, and ZIP Code*)		10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.

. TITLE (*Include Security Classification*)
HE AS FINANCIAL REPORTING SYSTEM: SOME EXPERIENCE ON PROTOTYPING AND USER INTERACTION

. PERSONAL AUTHOR(S)
onald L. Booker

a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (*Year, Month, Day*)	15. PAGE COUNT
aster's Thesis	FROM _____ TO _____	1986 March 27	211

. SUPPLEMENTARY NOTATION

COSATI CODES			18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)
FIELD	GROUP	SUB-GROUP	DSS, Prototypes, Adaptive Design

ABSTRACT (*Continue on reverse if necessary and identify by block number*)

daptive Design or prototypinghas been suggested as an effective approach for developing and mplementing computerbased decision support systems (DSS). The literature on DSS has also nown that adaptive design improves user involvement and satisfaction. This thesis attempts o follow the adaptive design approach to design and implements a dataoriented DSS--the PS Department of Administrative Sciences (AS) Database Reporting System. The development f the AS Reporting System has helped review the adaptive design strategy. The following indings were derived:

Close participation between end user(s) and the system builder helps to clarify problem ssues.

Frequent brief interactions between end user and system developer provide more immediate nd better defined reponse, resulting in a more effective prototype.

Unstructured initial planning sessions involving the end user and the system builder could istract the end user from the definition of important initial requirements. A more

2. DISTRIBUTION / AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION unclassified
a. NAME OF RESPONSIBLE INDIVIDUAL Tung Bui	22b. TELEPHONE (*Include Area Code*) 408 646-2630
	22c. OFFICE SYMBOL 54BD

FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Block #19

structured, issue-oriented approach results in more timely and clearly defined requirements.
-User feedback is less intense and critical towards the final phase of the development process.
-Prototyping is cost-effective, at least for the first phases of system enhancements.
-Prototypes are disposable systems.

Approved for public release; distribution is unlimited.

The AS Financial Reporting System: Some Experience on Prototyping and User Interaction

by

Ronald Lloyd Booker
Captain, United States Marine Corps
B.S., United States Naval Academy, 1977

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
March, 1986

Abstract

Adaptive Design or prototyping has been suggested as an effective approach for developing and implementing computer-based decision support systems (DSS). The literature on DSS has also shown that adaptive design improves user involvement and satisfaction. This thesis attempts to follow the adaptive design approach to design and implements a data-oriented DSS -- the NPS Department of Administrative Sciences (AS) Database Reporting System. The development of the AS Reporting System has helped review the adaptive design strategy. The following findings were derived:

- Close participation between end user(s) and the system builder helps to clarify problem issues.
- Frequent brief interactions between end user and system developer provide more immediate and better defined response, resulting in a more effective prototype.
- Unstructured initial planning sessions involving the end user and the system builder could distract the end user from the definition of important initial requirements. A more structured, issue-oriented approach results in more timely and clearly defined requirements.
- User feedback is less intense and critical towards the final phase of the development process.
- Prototyping is cost-effective, at least for the first phases of system enhancements.
- Prototypes are disposable systems.

TABLE OF CONTENTS

I. INTRODUCTION

The purpose of this thesis is twofold. First, it designs, develops and implements a data-oriented financial decision support system for the Department of Administrative Sciences (AS) of the U.S. Naval Postgraduate School -- the AS Database Reporting System. Second, throughout the development of the system, this thesis seeks to derive some observations on the effectiveness of the adaptive, or prototyping, approach in building an end user-oriented computerized decision support system.

There are at least two factors that have motivated this thesis. First, the increasing need of managing complex financial data within the Administrative Sciences Department has triggered interest among the department managers to automate their financial system. A computerized system is expected to provide accurate and timely information handling. It can also generate fast and standardized reports which are prohibitively labor intensive if done with the manual system.

Second, the literature on Decision Support Systems (DSS) often advocates that the traditional approach for developing a data processing system is not appropriate for prototyping. In fact, a conventional development life-cycle consists of eight phases. These include: system feasibility, software plans and requirements, product design, detailed design, code, integration, implementation and operations and maintenance [Ref. 1]. Such an approach requires full knowledge of the nature of the problem. Furthermore, it has revealed to be costly in time and effort. Since DSS requirements constantly change during the development process, using the traditional life-cycle model can be risky.

This thesis is organized as follows. Chapter II discusses the adaptive design approach to DSS development. Characteristics of the prototyping approach are addressed. Also, the model for prototyping is presented. Chapter III gives an account of the prototyping experiment with the AS Database Reporting System. The "diary approach" is adopted to record the evolution of the system architecture from the user interaction perspective. Chapter IV discusses some observations based on the prototyping experiment and provides suggestions for possible future extensions of the proposed system.

Three phases of iterative design are examined. Technical reports on the AS Database Reporting System are described in the appendices. Appendix A reproduces the user's manual. It provides instructions for the casual users. Diagrams of the system interface, reproduction of the system menus, and sample hard copy reports are respectively given in Appendices B, C and D. Appendix E contains detailed technical instructions intended for the database administrator and the system programmer. Data Flow Diagrams, structured charts, a data dictionary, program listings and lists of tables are respectively given in Appendices F, G, H, I and J.

II. THE ADAPTIVE DESIGN APPROACH TO DSS DEVELOPMENT

A. WHAT IS ADAPTIVE DESIGN?

Conventional design strategy in data processing systems invariably assumes that the requirements of the system can, and should, be determined prior to the implementation of the system. This requirements analysis can be achieved by performing logical analysis and investigation of user information processing needs and behavior [Ref. 2]. For instance, the requirements for an accounting system could be determined by examining accounting procedures and interviewing competent accountants.

However, such a procedure cannot be applied to building a DSS. A DSS can be defined as a computer-based system that helps its user(s) in solving ill-defined or unstructured decisions. As a consequence, the system designer cannot--and eventually should not -- have or be required to have a complete understanding of the user's needs prior to the design and implementation process. Rather, he should expect that the emergence of unanticipated informational needs is a continuing part of the design and development effort.

Thus, adaptive design in a DSS involves the role of learning in the DSS building process. To reduce the risk of not being able to reach (uncertain) final objectives, DSS researchers often argue for adopting the prototyping strategy. This consists of focusing the development effort on building a quick and working prototype or model that has the most important features or ideas. This early system is next delivered to the end user for evaluation. The users then give their feedback for determining possible enhancements. The system builders study the users' suggestions and attempt to update the prototype accordingly. This interaction

between the user and the analyst continues until a satisfactory prototype is achieved. An important implication of such an iterative and evolutionary process is that both the user and the analyst are expected to make mistakes, but attempt to learn as much as possible from these mistakes.

B. CHARACTERISTICS OF ADAPTIVE OR EVOLUTIONARY DESIGN

The process of evolutionary design can be characterized by the following:

1. Remove the Pressure to Do it Right

Keen [Ref. 3] insists that the prototyping goal should be focused on effectiveness rather than on efficiency. It is essential that a system be built, even if it does not have all of the desired features. Such a strategy will help remove some pressure on the system builder. In fact, high requirements of the system reliability could be a gold-plating feature that causes delay and increased costs. The end user often wishes to see a working system -- even at the risk of incompleteness -- to understand what it is all about.

2. A Learning Process

Learning is an integrated part of the design process. It is assumed that the user copes with a semi-structured domain. Some relevant decision-making factors are unknown. Also, the decision making methods are vaguely defined. Prototyping can be used as a learning tool to explore the manner of change in the users' priorities. A good development process should lead to a redefinition of the user's way of analyzing the problem. This would in turn lead to a new way to resolve problems.

3. User Participation

Evolutionary design involves active user participation. The user should provide good and timely feedback to facilitate the enhancement process of early prototypes.

This implies that the DSS user employs the system and evaluates it in light of the merits of the information it provides to support decision making. He is expected to suggest changes in Input/Output formats and the structure of the prototype under development, rank the usefulness of various modules of the current system and to prioritize the usefulness of suggested features to be integrated.

4. Inexpensive Development Effort

Keen and Gambino [Ref. 4] argue that the first prototype should be inexpensive. By its nature, prototyping is a risky venture. Since potential benefits of the would-be DSS are often unclear, important development cost can hardly be justified. More important, a high initial investment can discourage the user to abort the process.

C. A MODEL FOR PROTOTYPING OR ADAPTIVE DESIGN

Neumann and Jenkins propose a prototype model that consists of four procedures. As shown in Figure I, the first procedure is a design step that attempts to identify the user(s)'s basic information requirements. A number of requirements analysis approaches were proposed to identify the users(s)'s requirements; the data abstraction approach-- the essential features are data and data relationships, and the process simulating approach -- both data and processes must be identified in the first step.[Ref. 5]

The second step involves the development of the initial or enhanced prototype which can then be delivered to the user for use and feedback. In the third step, the use of the prototype leads to discussions and suggestions for enhancing the prototype (the fourth step).

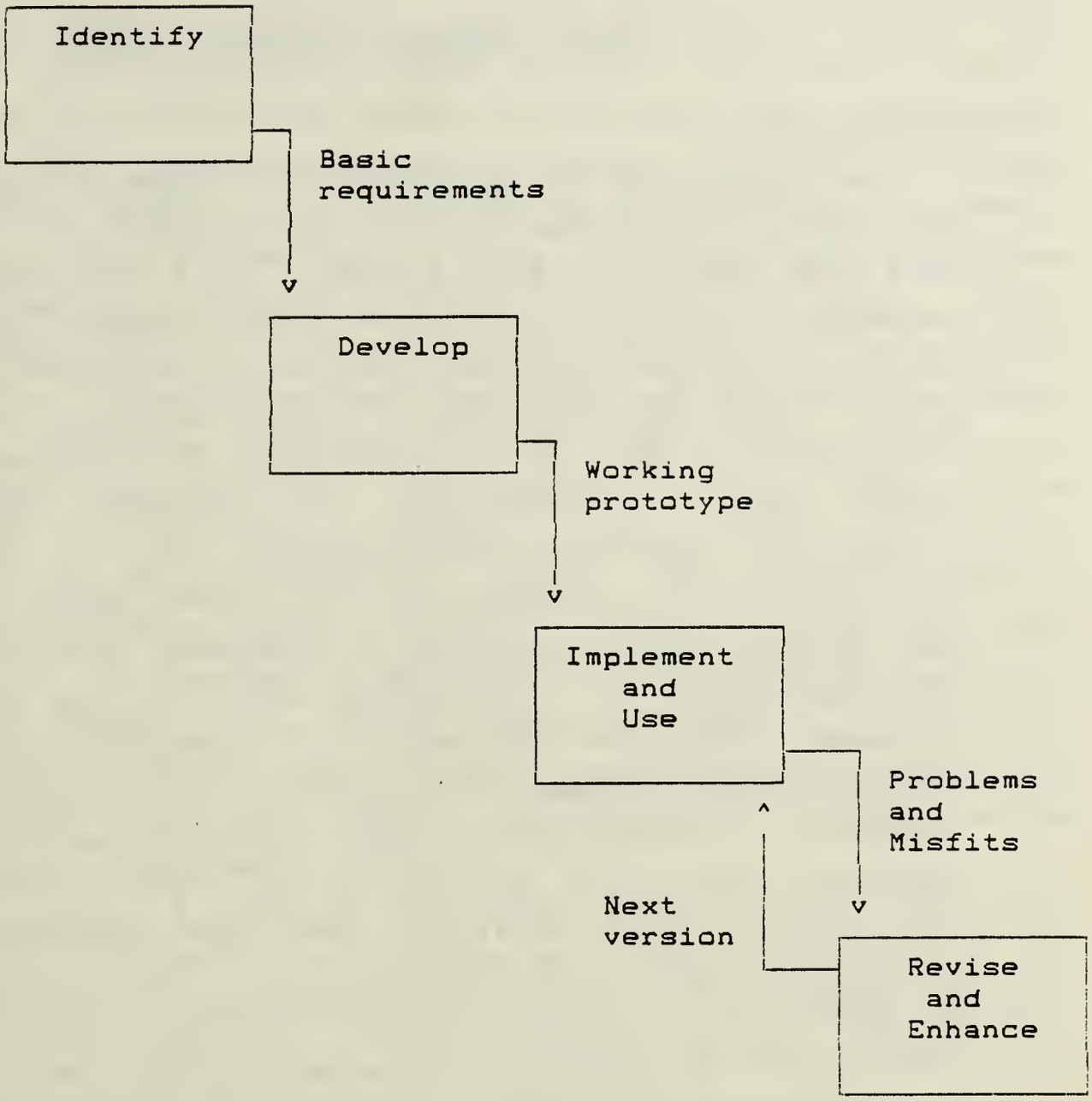


Figure 1
A Model for Prototyping

It should be noted however that the interaction between the user and the model is the key issue during the entire process. It is the quality and speed of the interaction between the user and the system developer that dictates the successfulness of the prototyping project.

III. THE AS DATABASE REPORTING SYSTEM

A. BACKGROUND

The Department of Administrative Sciences (AS) of the U.S. Naval Postgraduate School has operated its financial data base manually. With a size of more than one hundred personnel, including faculty and staff, financial operations and planning has required substantial investment in manpower. With its growing size, there has been a universal recognition to have an computer-based data base management system (DBMS). In March 1984, an information requirements analysis had been explored by Renner [Ref. 6]. This analysis applied the methodology of structured design to derive some suggestions for building a relational DBMS for the AS department [Ref. 7]. In November 1984, a first attempt to implement a DBMS was initiated by a Management Information Systems faculty member (who played the role of the system builder) in collaboration with a departmental management assistant (who played the role of the potential user). The KnowledgeMan (K-Man) relational Data Base Management System was adopted as the system generator. K-Man was chosen because of its flexible relational database concept and the possibility to perform spreadsheet analysis and graphics. It also leaves room for the trained end user to manipulate the DBMS command language. Extensive interaction between the two interested parties has lead to a thorough requirements analysis and system specifications. Also, a few data tables and displayed screen interfaces were developed to test the feasibility of using K-Man as the system generator.

B. METHODOLOGY: THE "DIARY APPROACH"

Ginzberg argues for a "Diary Approach" to observe the relationship between the user and the system designer [Ref. 8]. This approach has proven useful in recording the evolution of the systems modifications and user satisfaction. The diary approach consists of using a log book to systematically record the history of the DSS development process. Its records the evolution of the system due to modifications requested by both the system builder and the users. Figure 2 describes the three elements involved in the adaptive design of the AS database reporting system. Among the users, the management assistant and the faculty advisor maintained close interaction with the graduate student. The interaction with the faculty advisor primarily dealt with conceptual designs and technical issues. The interaction between the management assistant and the graduate student was almost daily. It focused on the system development and the interface format; i.e., screen design and reports. The interaction with the administrative assistant and interested office clerks was casual. They provide feedback on the quality of the user interface.

Table I reports historical entries on the iterative design process. The changes were either initiated by the user (i.e., the management assistant) or the system developer (i.e., the author). The definition of the acronyms used in the table can be found in Appendix H.

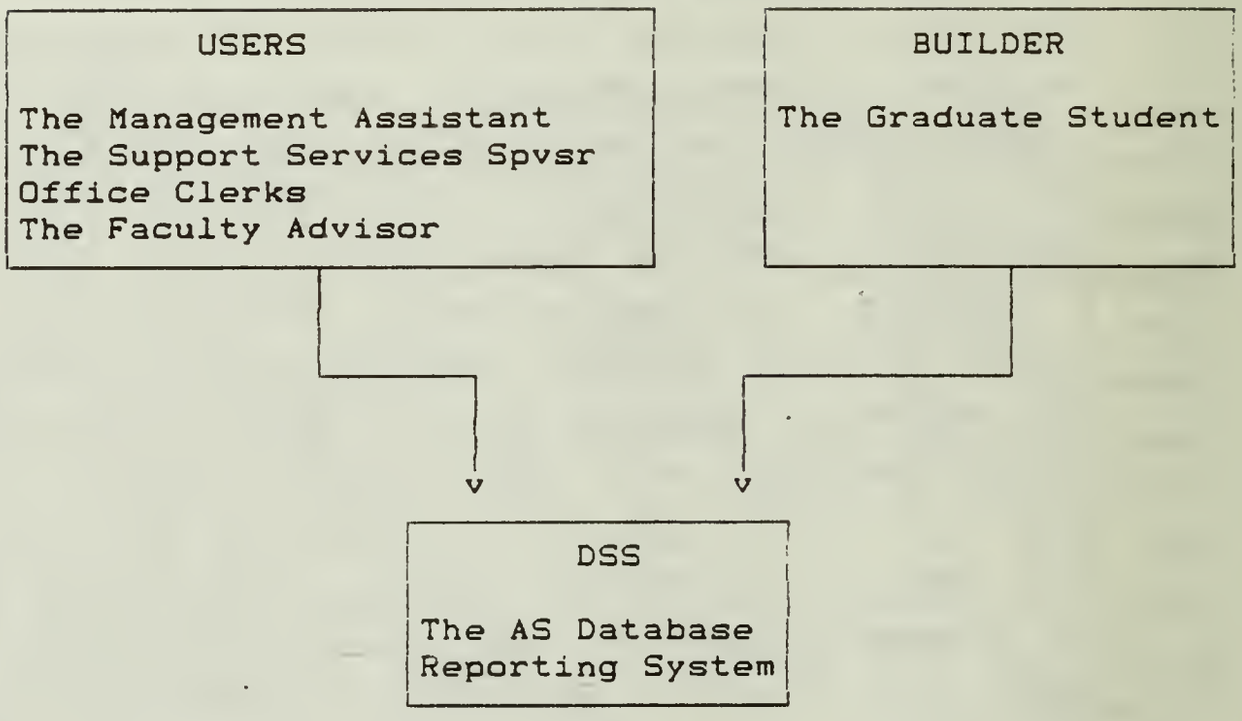


Figure 2 : The Elements of the Adaptive Design of the AS Database Reporting System

TABLE I. DIARY ENTRIES

DATE	INITIATOR (D = DEVELOPER, U = USER)	ACTION
3 Sep	D	Worked on forms to add data to the personnel history (PHISTORY) tables.
4 Sep	D	"
5 Sep	D	Same as above. K-Paint directions were misread; caused 2 days delay. Created three PHISTORY reports for user to confirm design.
8 Sep	D	Continued working on PHISTORY reports.
9 Sep	D	"
26 Sep	U	User had problems accessing files. This was caused by a hardware problem. Created three menus; the main, expenses and data base menus.
29 Sep	U	Modified/created 5 reports.
30 Sep	D	Modified the PHISTORY menu program and created a program to delete records.
3 Oct	D	Transferred files to another computer due to hardware problems. (Backed-up and reconfigured system.)
8 Oct	D	Everything is at the same state as 2 weeks ago - lost 2 weeks due to Backup problems.
15 Oct	D	Attempted to pass an ICF program in the deletion modules. -- Not successful.
15 Oct-13 Nov	D	Attempted to get around above problem by using MACRO(s). -- Not successful. Decided to use case statements if necessary in modules. -- This was successful.

13 Nov	D	Corrected PHISTORY report. Page headers were not displayed. Error was in reading K-Paint documentation.
15 Nov	D	Attempted to speed-up processing by RELEASING modules. -- Not successful.
18 Nov	D	RELEASE and MACRO(s) are not needed in the code.
20 Nov	U	Trying to determine a method of printing the last date that a table was updated on a report. -- Not successful.
21 Nov	D	Demonstrated system for user.
26 Nov	U	Made the system more user friendly (error detection) by using loops in programs.
1 Dec	D	Working on menus for the reports generators.
2 Dec	U	Modified PHISTORY report.
11 Dec	U	Created Outstanding TO(s) Report.
12 Dec	U	Modified forms for entering data from mixed case to upper case. Added a RATE field to PHISTORY table to ease computation of virtual fields.
14 Dec	D	Created forms to enter data and modified programs to display menus.
7 Jan	D	Modified programs to load tables and forms once to speed up processing.
9 Jan	D	Created form to enter data.
14 Jan	D	7 Jan modifications were not successful. Deleted modifications.
15 Jan	D	Demonstrated system to user. Implemented system.

18-26 Jan	U	Modified and created programs to allow user to access specific records. Required creation of 83 more files. (perform files, forms and error messages)
3 Feb	U	Modified the personnel funding report.
6 Feb	U	Deleted INDIRECT.DOC# from INDIRECT entry forms and added SEGMENT and SERIALS to INDIRECT table.
	D	Modified the DELTREC2.IPF module so that the last record in a table will not be deleted if a match of the deletion criterion is not found.
13 Feb	U	Took INDIRECT.EXTENDED out of tables -- not needed.
16 Feb	U	Added PURCHASE.QTY to entry forms.
17 Feb	U	Added INDIRECT.SERIALS and INDIRECT.SEGMENT to entry form. Deleted INDIRECT.DOC# from same.
19 Feb	U	Found two different data values called INDIRECT.DOC#. Placed INDIRECT.DOC# back in entry form.
	D/U	Took financial analysis option out of MAINFRM.IPF (not enough time to develop and implement). Crossed modified system over to production.

With the "Diary Approach" any major design decisions or accomplishments, and significant problems or events were logged. As the design of the system progressed, the majority of the actions were initiated by the user(s).

C. THE EVOLUTION OF THE AS REPORTING SYSTEM ARCHITECTURE

Departing from the original architecture, this section discusses the required modifications that occurred throughout the implementation and testing processes. During the development process, it has been noticed that the user has actively participated in the creation of the DBMS prototype.

Such participation has triggered numerous generations of new requirements and consequently modifications to the originally planned design. The purposes of this thesis are (i) to expand the initial system, a working but incomplete data base system that is used as a management planning tool, and (ii) to observe the evolutionary development of the DBMS in terms of user needs and satisfaction.

1. The Initial AS Reporting System Architecture

The initial AS Reporting System Architecture (Figure 3) was based on four income accounts; OPTAR, 10% Indirect, other sources of income and research funds, and four expense accounts; miscellaneous expenses, travel, personnel salary and equipment/supplies. Figure 3 shows that the income accounts had varied expenses; i.e., miscellaneous expenses, travel and equipment/supplies may be funded by the OPTAR account. Additionally, there may be major restrictions on expense funding; i.e., equipment/supply purchases over $3000 may not be funded by the OPTAR account, personnel salaries may not be funded from the OPTAR account.

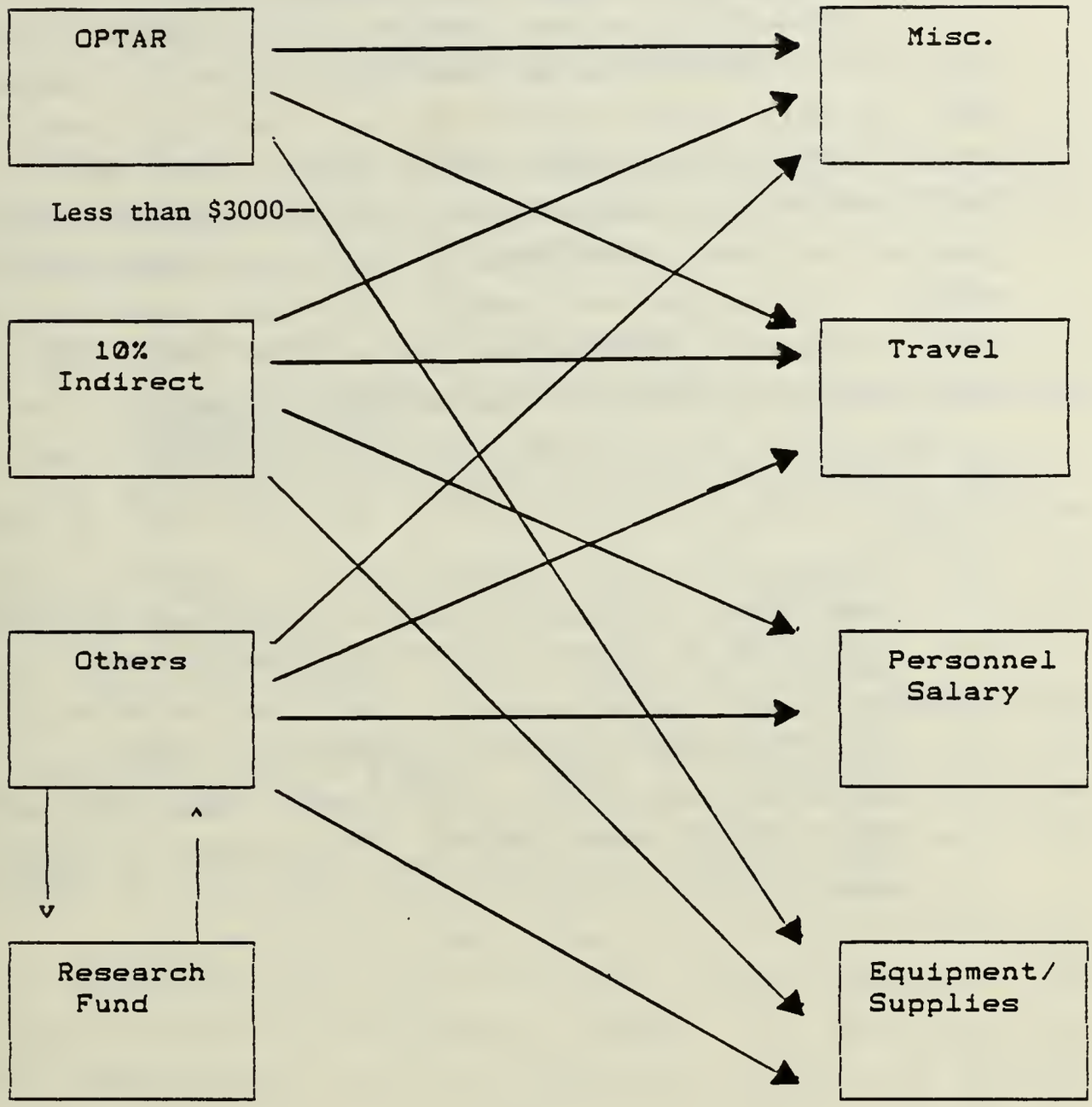


Figure 3: The Initial AS Reporting System Architecture

2. The First Modification Stage

During the first modification stage, an additional income account, OPTAR balance, and an additional expense account, travel request, were deemed necessary to provide the user(s) with more financial information.

3. The Second (Final) Modification Stage

During this stage, accounts (data tables) were added and deleted from the AS Reporting System's Architecture. The two data tables added during the first modification stage, OPTAR Balance and travel request were deleted. The OPTAR and travel accounts, with minor modifications, could provide the same information as the before mentioned deleted tables. Two tables were added, one table contained personnel information (pay grade, job status, etc.) and another contained an inventory of property. Appendix J is a list of the current data tables and their respective data fields. The following are major changes resulting from user interaction:

1. Redefinition of field names and types to increase consistency throughout the data base and deletion or addition of data fields; redefined fields -- 7, deletion of fields -- 12, addition of fields -- 16.
2. Menu and form formats.
3. Change of instructions on menus and forms.
4. Compromise of report formats to fit both users; the more knowledgeable user wanted more information while novice users wanted less.
5. The ability to select records based on specific field values.

IV. RESULTS AND DISCUSSION

This thesis involved the design and implementation of a data-oriented decision support system to support financial decision making of the AS Department.

A. SOME OBSERVATIONS ON THE EXPERIMENT

1. Close Participation Between the End User(s) and the Systems Builder

Adaptive Design allows the user to actively participate in the evolution of the system characteristics. Three basic changes in the system design occurred during the prototyping effort. First, some data tables were redesigned. Field names and sizes had been redefined. A number of tables were combined. Second, additional program modules were added to coordinate various systems functions using menu structures. The latter render the sequencing of program modules transparent to novice users. Third, the use of the system (i.e., K-Man) command language was increased.

In a more conceptual level, the cross examination of the prototype give rise to better understanding of the issues related to the financial system. The experience with the implementation of the AS Reporting System also confirms some early findings that argues prototyping as a catalyst to participative design, learning, and organizational development [Ref. 9]. In effect, it has been observed that the main participating user has gained increasing interest in the understanding and design of systems. Also, the early success of the prototype has generated some redesign of office tasks.

2. Interaction Time

The more frequent the interaction time between the user and the developer, the more effective the prototype.

Delayed feedback has proven to slow down the learning process and weaken the participants' enthusiasm. Time pressures could be used as a strategy to speed up the interaction process.

3. Planning Sessions

Unstructured initial planning sessions involving the end user and the system builder could distract the end user from the definition of important initial requirements. A more structured, issue-oriented approach results in more timely and clearly defined requirements. This is even accentuated when there is more than one participating user and/or novice users. Individual differences sometimes have produced conflicting design suggestions. However, effective group discussion have proven to avoid the risk of designing system features that reflect strong biases from a single individual. Henderson [Ref. 10] raises the potential risk of loss of creativity when dealing with multiple participating users. Due to little group interaction, and more importantly, to the large disparity in the degree of participation, it was not possible to confirm nor disconfirm the impact of group interaction processes and user creativity.

4. User Feedback is Less Intense and Crucial Towards the Final Phase of the Development Process

The user feedback is expected to end when a satisfactory system is delivered. It seems to be unrealistic to expect the end user -- even though he/she has become more computer literate -- to keep expanding the system. Since the user has captured the capabilities as well as limitations of the system, he/she can more or less precisely formulate requirements for future enhancements. At this

stage, the need of sustained user feedback seems to be less crucial -- requested modifications are normally minor.

5. Cost Effective

Prototyping is cost effective, at least for the first phase of system enhancements. The effort spent on developing a successful and complete module has allowed duplication of programming procedures to similar modules in later version of the prototype. However, it is not clear that the economies of scale effect would continue to implement more complex integrated modules in the future.

6. Prototypes are Disposable Systems

At least for the user who has participated throughout the entire design and implementation process, the prototype has become to a certain degree disposable. Since the participative user has become increasingly computer literate, some of the initially requested DSS features have revealed to be less useful. In fact, the ease-of-use and structured features of the system has reduced the DSS to perform only for a well-defined set of data manipulation and reporting. To perform unusual data manipulation, a trained user can now directly use the DSS generator command language. However, this seems to be true only for the user who has worked closely with the development of the system. Other users, who are primarily casual users and non computer experts, are expected to find the system useful.

B. RECOMMENDATIONS FOR FUTURE EXTENSIONS

By definition, the adaptive design should help the system to be progressively expanded. The current system primarily handles data management and report generation. There are a number of possible enhancements that could be added to the current release of the AS database reporting system.

1. Forms Handling

The AS department office information system operates via a multiplicity of predesigned paper forms such as travel orders, purchasing stubs, reimbursement forms, etc.. These forms have been processed by office clerks using typewriters. Additional report procedures could be added to the current release of the AS Database Reporting System to generate these stub from the computer printers. Such a capability would significantly improve the office productivity by reducing typing errors, and duplication of effort in twice entering the same set of information using both the typewriter and the computer.

The generation of office forms for the printer can be done by using the K-Report report generation feature to replicate forms. This could be done without much difficulty. It should be noted however that continuous customized computer forms must be ordered.

2. Financial Planning

To support financial planning, data that reflect the balances of various expenses accounts should be rapidly computed. Also, the user should be able to import selected data from data tables to handle spreadsheet analysis. The use of spreadsheets would allow the user to perform what-if analysis on various financial decisions. Furthermore, a trend analysis can be supported by the system by displaying various graphical charts of key financial figures. To implement the financial planning capability to the system, the following steps are suggested:

1. Join relevant financial data tables together.
2. Write procedures to compute conditional aggregate values.
3. Import aggregate values into the K-Man spreadsheet.
4. Describe procedural steps for the user to manipulate spreadsheet functions.
5. Import data from the spreadsheet to the K-Graph feature for graph generation.

3. Regulation Engineering

As with any governmental agency, the financial decision making process has to strictly follow the laws and regulations determined by the Federal Government. These regulations include travel expenses (e.g., 10% money cannot be used for Invitational Travel Order), personnel (e.g., the OPTAR account cannot be used for personnel salaries), and equipment purchases (e.g., sole source justification is needed for expenses whose amount is more than one thousand dollars; purchase using OM & N funds cannot exceed three thousand dollars). The responsibility to make sure that financial decisions are legal relies solely on the user. Due to the complexity and diverseness of regulations, it would be useful that the system can store some knowledge in the system. The latter can be used as a internal comptroller that instantaneously checks the legal aspect of all financial transactions.

The knowledge on various legal issues and financial practices could be acquired and integrated into the current AS database reporting system by using GURU; an off-the-shelf expert system development tool and a by-product that can be fully interfaced with K-Man.

V. CONCLUSIONS

The purpose of this thesis was twofold:

1. To use the adaptive or prototyping approach to implement a data-oriented DSS for tracking financial transactions of the Naval Postgraduate School's Department of Administrative Sciences.
2. To derive from the system development, some observations on the impact of the prototyping approach on systems design.

These two objectives have been achieved. First, the AS Database Reporting System has been used since January 1986. The system has gained in usage as the system has benefited from enhanced capabilities. Also, the number of systems users has also increasing. Travel clerks and purchasing clerks begin to use the system on a regular basis. Various members of the department also use the system on a casual basis via the Management Assistant who is currently the database administrator. Second, this thesis confirmed some of the early results found recently on the DSS literature regarding the DSS design strategy.

The results of this study was reported both in the main body of the thesis and the appendices. Chapter I described basic concepts and issues relating to prototyping. Chapter II reported the development process of the AS Database reporting system in light of the adaptive design. The "Diary Approach" was adopted to record development activities and the interaction between the user and the system developer. During the development process, close interaction with the user has lead to three modifications of the AS Reporting System Architecture. Chapter III analyzed and discussed the results of the experiment. Some suggestions for future extensions of the current database were also presented. The integration of spreadsheet analysis, graphical displays, and

a knowledge-based system to the current version of the AS Database Reporting System would greatly improve the decision support role of the implemented financial system.

VI. APPENDIXES

APPENDIX A
ADMINISTRATIVE SCIENCES (AS)
DATA BASE REPORTING SYSTEM

USER'S MANUAL

TABLE OF CONTENTS

I. INTRODUCTION

The purpose of this user's manual is to guide the first or casual user to get acquainted with the operations of the Administrative Sciences (AS) Data Base Reporting System (the AS System). The current version of the AS System is an interactive, menu driven data base management system. This system is used to input, edit and store the U.S. Naval Postgraduate School's Administrative Sciences Department's financial data (i.e., the AS Department's OPTAR, data relative to the AS Department's indirect cost and research account, and the Department's equipment inventory) and to create financial reports for budgetary planning. Because this is a menu driven system, prior programming knowledge is not required to use the AS System.

For the sake of simplicity, aspects that are related to the conceptual design and technical specifications of the AS System are described in separate documents -- the Programmer's Manual and the thesis titled "The AS Financial Reporting System: Some Experience on Prototyping and User Interaction"; written by R.L. Booker.

This manual is organized as follow. General and most frequently asked information are provided in chapters I through III while more specific and less used information are provided in chapters IV and V.

II. PRINCIPAL USERS

The principal users of this system are expected to be the AS Department's Support Services Supervisor, Management Assistant, Travel Clerk and Procurement Clerk. The Management Assistant is the AS System's Data Base Administrator (DBA). All request or suggestions for changes and major problems should be referred to the Data Base Administrator.

Table I indicates some functions that the AS System can currently provide to support the tasks of each of the above mentioned users.

USERS	DUTIES	SYSTEM SUPPORT
Support Services Supervisor	Supervises, plans and coordinates the department's support functions and clerical support.	Status update on personnel.
Management Assistant	Interprets reports of all financial receipts, expenditures and records. Identifies contributing sources and makes comparisons to establish budget trends.	Updates financial balances from purchase and travel expenses reports.
Travel Clerk	Prepares and manages all travel related documents for department. Coordinates and manages all travel related arrangements.	Outstanding TO(s) report
Procurement Clerk	Receives requests for department procurement.	Requisition Status and Inventory reports

TABLE I. SYSTEM FUNCTIONS

III. OPERATION OF SYSTEM

A. LOCATION

The System is resident on the IBM-PC XT in room Ingersoll 230.

B. START-UP (BOOT-UP)

1. Place the provided DOS version 2.1 diskette in floppy drive (slot) A of the PC; the left drive in Figure 1.
2. The computer and printers are connected to a toggle switch under the computer table. The on-off switch on the computer should always be in the on position. Turn the toggle switch on to turn the computer and printer on. The computer will run its system's checks. (Steps 1 and 2 are considered a cold start) Note the warm start procedures below.
3. When the C> prompt appears, type in KMAN MAINMEN.
4. When prompted, type in your name and password.
5. When the AS System menus appear, make your menu selections.

Warm Start Procedures:

If the power has been previously turned on and the provided DOS version 2.1 diskette has been previously loaded, steps 1 and 2 of the start-up instructions may be ignored. Instead, together press the two keys marked CONTROL and ALT, and then the key marked DELETE (This is considered a warm start.). Now perform steps 3 through 5.

C. EXIT

1. Select the last option in the main menu to exit the AS System.
2. Turn the toggle switch off. (power off)
3. Take the diskette out of the computer.

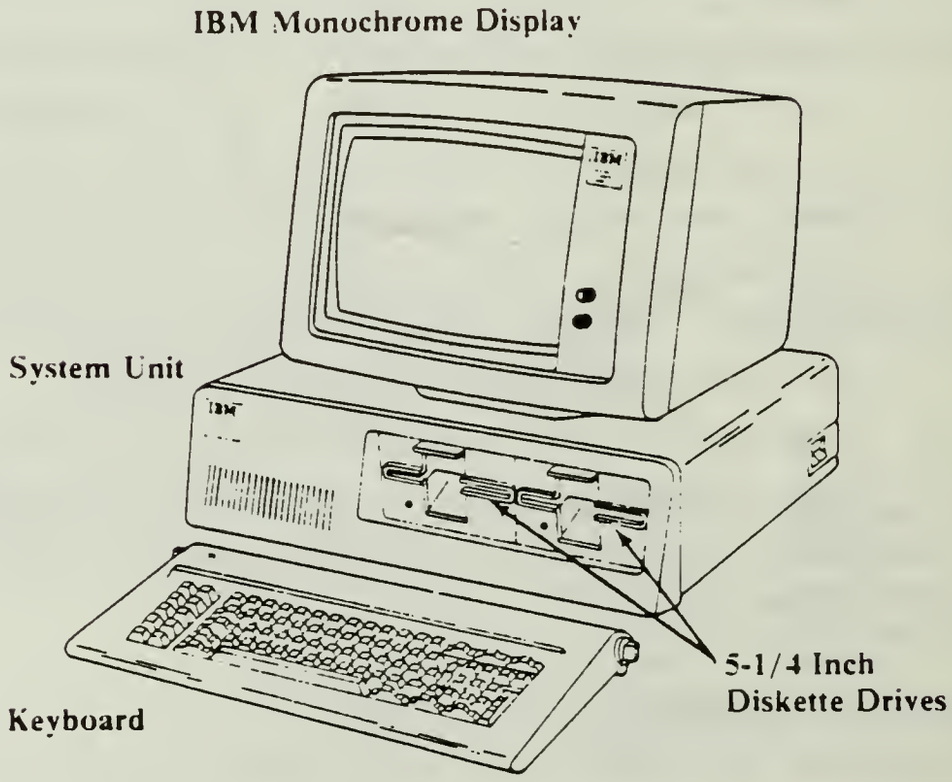


Figure 1. IBM PC

D. DATA MANIPULATION

All of the table manipulation instructions are displayed on BLUE menus. The following options are available for each table:

- browse a table	: allows viewing of all the records in a table
- update a record	: allows modifications of the contents of records given specific selection criterion
- add a record	: allows the addition of a record to a table
- delete a record	: allows the deletion of a record from a table based on specific criterion
- return to calling menu	: will display the next higher level menu

Appendix B is a chart of the AS Systems menu's hierarchy and Appendix C displays the AS System's Menus.

1. To View All of the Records in a Table.

Select the "browse a table" option in the specific table menu. A form will appear displaying the first record within the table. The table is sorted based on a predetermined requirement. The bottom of the form contains instructions for moving the cursor on the form and for viewing the prior or next record. If an attempt is made to display pass the last record in a table, the cursor will move to the BOTTOM LEFT corner of the screen and the table's menu will be displayed.

Note -- Pressing ESC in the BOTTOM LEFT corner below.

2. To Update a Record or Group of Records in a Table.

Select the "update a record" option in the specific table's menu. A form will appear requesting the entry of a

specific value that will select only the records containing that exact value. Enter the selection value.

a. If a record meets the selection value criterion, the only record, or the first record, that meets the selection criterion will be displayed on a form. The bottom of the form contains instructions for moving the cursor on the form and for viewing the prior or next record that meets the selection criterion. If an attempt is made to display pass the last record that meets the selection criterion, the cursor will move to the BOTTOM LEFT corner of the screen and the table's menu will be displayed.

b. If a record does not meet the selection criterion, the following message will be displayed - "No records satisfy this request". The table's menu will then be displayed. Confirm that the selection value was entered or spelled correctly by selecting the "browse a table" option.

Repeat the steps to update a record. If the problem continues, notify the AS System Administrator.

Note - Pressing ESC in the BOTTOM LEFT corner below.

3. To Add a Record.

Select the "add a record" option in the specific table's menu. A form will appear in which the data for a specific record is entered. The form contains instructions for moving the cursor on the form.

Note - Pressing ESC in the BOTTOM LEFT corner below.

4. To Delete a Record in a Table.

Select the "delete a record" option in the specific table's menu. A form will appear requesting the entry of a selection value that will enable the display of only the record(s) containing that selection value. Enter the selec-

tion value. Another form will appear confirming deletion of a record. If Y is entered, you are committed to deleting a record. If N is entered, the table's menu will be displayed and no record(s) will be deleted.

If Y is entered for deleting a record and a record meets the selection value, the only record, or the first record, that meets the selection criterion will be displayed on a form. The bottom of the form contains instructions for moving the cursor on the form and for viewing the prior or next record that meets the selection criterion. Place the cursor in the TOP LEFT corner of the form and press the ESC key to delete a record. ONLY the record that is displayed when the ESC key is pressed will be deleted. After the record has been deleted, the table's menu will be displayed.

Note - Pressing ESC in the BOTTOM LEFT corner below.

WARNING

If more than one record meets the selection criterion, all of the records that meet the criterion will be able to be displayed and deleted. If you attempt to display pass the LAST record that meets this criterion, this record will be DELETED without the requirement to press the ESC key. Therefore, check the displayed record to ensure that it is not the record for deletion before attempting to display the next record. Remember, the LAST record that matches your selection criterion will be DELETED if displaying pass the LAST record is attempted.

Pressing ESC in the BOTTOM LEFT Corner.

If ESC is pressed while the cursor is in the BOTTOM LEFT corner, the following message will appear: FEXIT or FABORT hit - are you sure (Y/N) ?

- Y will exit the AS System.
- N will display the table's or report's calling menu.

E. PRINTING PREDEFINED REPORTS

For the current version, five reports have been generated and could be automatically invoked. These include the Quarterly Inventory, Outstanding TO(s), Personnel Funding, Personnel History and the Requisition Status reports. Appendix D contains samples of the AS System's Reports. The menus for REPORT printing are colored BROWN. Select the menu containing the report that is required to be printed. ENSURE that the printer is CONNECTED to the computer and turned ON. Select the option for the report that requires printing. (Due to numerous sorts involved in printing of the Personnel Funding Report, there is a 1-3 minute wait before the report is printing.) After the report has been printed, the calling menu will be displayed.

Note -- If printing of a report is attempted while the printer is not on or connected to the computer, an error message will be displayed. Connect the printer and turn it on. Now attempt to print the report.

Note -- All reports are formatted for the NEC 3550 printer. Use of other printers will require changes to the AS System's programs. Notify the DBA if this is necessary.

IV. TABLES

The AS System's data is stored in the following tables:

1. Indirect - information on research and other reimbursable sources of funds and resulting Indirect Costs to Department
2. OPTAR - contains information on Department's operating budget
3. Other - contains information on other sources of Department funding
4. Personnel Funding - contains personnel expense data
5. Personnel History - contains personnel data for staff positions within the AS Department
6. Purchase - contains data on Department and research purchases
7. Travel Expenses - Department and Research travel expenses

A. SOURCES OF DATA FOR TABLES

Indirect Table:

- entry date - date record entered into the AS System
- status - if research, **pending** when first entered into AS System and changed to **funded** when funding document received from Comptroller
  - if other reimbursable, **funded**
- principal - if research, last name of principal investigator on proposal
  - if other reimbursable, from Code 002 Reimbursable Funds Status Report

- sponsor - if research, from proposal
  - if other reimbursable, **source** from Code 002 Reimbursable Funds Status Report
- proposal - if research, actual title of proposal (confirm with funding document when funding document received)
  - if other reimbursable, **title** from Code 002 Reimbursable Funds Status Report
- period - if research, dates specified on proposal for performance of research
  - if other reimbursable, no entry
- estimate - if research, estimated cost of research for current fiscal year from proposal's budget page
  - if other reimbursable, no entry
- DEPTIC - if research, the amount that the department is expected to receive in indirect cost; calculated from the proposal's budget page (10% of total labor charges)
  - if other reimbursable, no entry
- MIPR - if research, **MIPR #** from Block #5 of funding document (Military Inter-department Purchase Request)
  - if other reimbursable, no entry
- ref# - if research, **Reference #** from Block #5 of funding document (Order for Work and Services, NAVCOMPT 2275)
  - if other reimbursable, no entry
- authorized - if research, **actual authorization** from Code 002 Memorandum (Funds Authorization Grant/ Change)
  - if other reimbursable, **act auth** from Code 002 Reimbursable Funds Status Report

- costcode
  - if research, **job order** from Code 002 Memorandum (Funds Authorization Grant/Change)
  - if other reimbursable, **JO #** from Code 002 Reimbursable Funds Status Report
- expiration
  - if research, **exp date** from Code 002 Memorandum (Funds Authorization Grant/Change)
  - if other reimbursable, **exp date** from Code 002 Reimbursable Funds Status Report
- ICRECD
  - if research, **Indirect (Chairman)** from Code 002 Memorandum (Funds Authorization Grant/Change)
  - if other reimbursable, **ind cost (chair)** from Code 002 Reimbursable Funds Status Report
- staff labor
  - if research, total of all support labor on proposal's budget page
  - if other reimbursable, no entry
- as appropriation
  - if research, amount which the principal investigator agreed to give the Department
  - if other reimbursable, verbal authorization from principal investigator or Department Chairman.
- doc#
  - if research, **Document #** from Block #2 of funding document (Order for Work and Services, NAVCOMPT 2275)
- segment
  - if research, **segment** from Code 002 Memorandum (Funds Authorization Grant/Change)
  - if other reimbursable, **seg** from Code 002 Reimbursable Funds Status Report
- serials
  - if research, **serial #'s** from Code 002 Memorandum (Funds Authorization Grant/Change)
  - if other reimbursable, **doc #'s** from Code 002 Reimbursable Funds Status Report

OPTAR Table:

- entry date - date record added to AS System
- date - date of latest authorization memo received
- authorized - total amount authorized; from Code 00 Memorandum (Fiscal Year Interim Operating Target)
- travel authorized - amount designated for travel; from Code 00 Memorandum (Fiscal Year Interim Operating Target)
- actual - changes in authorized; from reissues of Code 00 memorandum (Fiscal Year Revised Interim Operating Target)
- travel actual - changes in travel authorized; from reissues of Code 00 Memorandum (Fiscal Year Revised Operating Target)
- difference - (actual - authorized); computed by AS System
- travel diff - (travel actual - travel authorized); computed by AS System

Other Table: - comes from verbal authorization of Department Chairman or Associate Chairman for Research and the principal investigator

- entry date - date that the record is entered into the AS System
- status - status of availability of funds to Department; **pending, received,** or **expired**
- expiration - **Expires** from Research Administration printout or **Exp Date** from Reimbursable Funds Status Report provided at time of agreement
- principal - **Account** from Research Administration printout; or **principle investigator** from Reimbursable Funds Status Report
- sponsor - **sponsor** from Research Administration printout or **source** from Reimbursable Funds Status Report
- proposal - **title** from Research Administration printout or Reimbursable Funds Status Report
- costcode - **cost code** from Research Administration printout or **JO#** from Reimbursable Funds Status Report
- authorized - amount agreed upon for Departmental use
- remarks - misc

Personnel Funding Table: all data comes from the department's outgoing Support Employment Schedule

- entry date - date of record entry to AS System
- last name - from **employee**, from outgoing Support Employment Schedule Memorandum
- first name - from Personnel History table
- fund date - date of outgoing Support Employment Schedule Memorandum
- fund from - date funding period commences, **dates** from outgoing Support Employment Schedule Memorandum
- fund to - date funding period terminates, **dates** from outgoing Support Employment Schedule Memorandum
- # of days - number of 8-hour days worked during funding period, **days** from outgoing Support Employment Schedule Memorandum
- hours - number of hours in excess of 8-hour days worked, **hours** from outgoing Support Employment Schedule Memorandum
- cost code - from outgoing Support Employment Schedule
- code - from the following department codes:
  - SS; support salaries
- amount - calculated based upon rate and number of days/hours worked
- rate - employee's rate during funding period, **rate** from personnel funding
- hours per week - number of hours employee worked during funding period, **hours per week** from personnel history

Personnel History Table:

- entry date - date record is entered into table
- last name - from block # 1 of Standard Form 50 (Notification of Personnel Action
- first - from block # 1 of Standard Form 50 (Notification name of Personnel Action)
- DOB - date of birth from block # 4 of Standard Form 50 (Notification of Personnel Action)
- effective - from block # 14 of Standard Form 50 (Notification of Personnel Action)
- status - from block # of Standard Form 50 (Notification of Personnel action); T = temporary; P = permanent
- billet # - from block # 27 of Standard Form 50 (Notification of Personnel Action); assigned by CPO
- job title - from block # 27 of Standard Form 50 (Notification of Personnel Action)
- elements/ stds set - date elements and standards set for performance review; from current performance appraisal
- work unit - assigned by support services supervisor, if any
- work - **immediate supervisor,** from Block # of Position director Description cover sheet
- PD # - from block # 27 of Position Description cover sheet or block # 27 of Standard Form 50 (Notification of Personnel Action)
- series - from block # 29 of Standard Form 50 (Notification of Personnel Action)
- grade - from block # 30 of Standard Form 50 (Notification of Personnel Action)
- step - from block # 31 of Standard Form 50 (Notification of Personnel Action)

- annual salary - from block # 32 of Standard Form 50 (Notification of Personnel Action)
- hours per week - # of hours worked per week, from Block # 37 of Standard Form 50 (Notification of Personnel Action)
- expire - date of expiration of temporary appointment, from block # 18B of Standard Form 50 (Notification of Personnel Action)
- rate - value automatically computed by the System
- remarks - misc

Purchase Table:

- entry date - date that the record is entered into AS System
- document # - from block # 36-43 of DD Form 1348 (Requisition System Document)
- costcode - or JO#, from block # 46-50 of DD Form 1348 (Requisition System Document)
- vendor - from block # A of DD Form 1348 (Requisition System Document)
- item description - from block # 8-22 of DD Form 1348 (Requisition System Document)
- stock # - or part #, from block # 8-22 or Remarks of DD Form 1348 (Requisition System Document)
- estimate - from block # T of DD Form 1348 (Requisition System Document)
- priority - from block # 60-61 of DD Form 1348 (Requisition System Document)
- RDD - required delivery date, from block # 62-64 of DD Form 1348 (Requisition System Document)
- code - from the following department codes:
  - LO; transportation of things (Federal Express)
  - MR; rental of equipment
  - PM; maintenance of minor property
  - PP; maintenance of plant property
  - QH; honoraria
  - QT; registration fees
  - TS; consumable supplies
  - WM; minor property acquisition
  - WMC; computer equipment/accessories
  - WP; plant property acquisition
  - WPC; computer equipment
  - YP; commercial printing

- requestor - last name of individual who requested purchase, from Block # of DD 1348 (Requisition System Document)
- PO# - purchase order # from Block # 2 of DD 1155 (Purchase Order)
- received - date item received, from receiving copy of DD 1348 (Requisition System Document)
- firm price - final price, from Block U of DD 1348 (Requisition System Document) or Block # 23 of DD 1155 (Purchase Order)
- serial - if a plant account item, from equipment
- pa# - plant account #, from inventory records forwarded by receipt control
- issue - name of individual plant account equipment issued to, from
- location - location of plant account equipment, from
- qty - from Block # of DD 1348 (Requisition System Document)
- remarks - misc

Travel Expenses Table:

- departure
  - if civilian orders, from Block # 10B of DD 1610 (Request and Authorization for TDY Travel of DOD Personnel)
  - if military orders, from Block # 9 of NAVPERS1320/16 (TEMADD Travel Orders)
  - if invitational orders, from **Proceed on or about** from NAVSO4650/10 (Invitational Travel Order)
- requester
  - if civilian orders, from Block # 2 of DD 1610 (Request and Authorization for TDY Travel of DOD Personnel)
  - if military orders, from Block # 3 of NAVPERS1320/16 (TEMADD Travel Orders)
  - if invitational orders, from Block # 1 of NAVSO4650/10 (Invitational Travel Order)
- location
  - destination of traveler
  - if civilian orders, from Block # 11 of DD 1610 (Request and Authorization for TDY Travel of DOD Personnel)
  - if military orders, from Block # 13 of NAVPERS1320/16 (TEMADD Travel Orders)
  - if invitational orders, from Block # 1 of NAVSO4650/10 (Invitational Travel Order)
- code
  - type of travel from the following department codes:
    - EI; invitational travel
    - ET; TAD travel
- estimate
  - estimated cost of travel
  - if civilian orders, **total** from Block # 14 of DD1610 (Request and Authorization for Travel of DOD Personnel)
  - if military orders, **total** from Block # 18 of NAVPERS1320/16 (TEMADD Travel Orders)

- if invitational orders, from **for the purpose of** from NAVSO4650/10 (Invitational Travel Order)

- costcode
  - or JO#, from travel orders
  - if civilian orders, from Block # 19 (62217-**XXXXX)** of DD1610 (Request and Authorization for Travel of DOD Personnel)
  - if military orders, from Block # 17(8) of NAVPERS1320/16 (TEMADD Travel Order)
  - if invitational orders, from Block # 7 (62271**XXXXX**) of NAVSO4650/10 (Invitational Travel Order)
- doc #
  - if civilian orders, **tango** from Block # 19 of DD1610 (Request and Authorization of TDY Travel of DOD Personnel)
  - if military orders, **tango** from Block # 4 of NAVOERS1320/16 (TEMADD Travel Order)
  - if invitational orders, from Block # 7 (ITXXXXX) of NAVSO4650/10 (Invitational Travel Order)
- advance
  - if civilian orders, from Block #15 of DD-1610 (Request and Authorization of TDY Travel of DOD Personnel)
  - if military orders, from Block # 21 of NAVPERS1320/16 (TEMADD Travel Order)
  - if invitational orders, no entry
- return
  - date of return, calculated from travel orders
- claim date
  - from Block # 14 of DD Form 1351-2 (Travel Voucher or Sub-voucher)
- firm price
  - total of Blocks #7 and # 10 from DD Form 1351-2 (Travel Voucher or Sub-voucher)
- entry date
  - date of record entry to AS System

- DOV # - **DO Voucher No.** from Block # 10 of DD Form 1351-2 (Travel Voucher or Sub-voucher)

Inventory Table:

- pa# - plant account number, from Block # 54 of DD 1342 (DOD Property Record)
- vendor - from Block # 14 of DD 1342 (DOD Property Record)
- description- from Block # 26 of DD 1342 (DOD Property Record)
- serial # - from Block # 17 of DD 1342 (DOD Property Record)
- PO# - purchase order number, from Block # 25 of DD 1342 (DOD Property Record)
- cost - from Block # 6 of DD 1342 (DOD Property Record)
- received - from Block # 54 of DD 1342 (DOD Property Record)
- issue - from Block # 28 of DD 1342 (DOD Property Record)
- location - from Block # 28 of DD 1342 (DOD Property Record)

V. ERRORS AND ABNORMAL PROCESSING

A. ERRORS

Notification of errors are given by two methods:

1. The AS System will display a WHITE on RED flashing message, which will prompt for a legal entry. When entering numbers, only use the numbers along the top of the keyboard.
2. The language (KnowledgeMan) or the Disk Operating System (DOS) will display a WHITE on BLACK message. The following are examples of messages which may be displayed:
   a. Do you want to escape (Y/N)? - Normally enter N (no).
   b. <type> error reading <device>; Abort, Retry, Ignore? - Normally enter R (retry).
      If display of the message continues, enter A (abort).
   c. A warning message may appear; example - File not found. Note the error message, exit the AS System, remove the diskette and notify the System Administrator.

B. ABNORMAL PROCESSING

If the AS System does not process normally (takes a longer than normal time to process), press the ESC key. Pressing the ESC key will either return processing to the calling menu or exit the AS System and the KnowledgeMan prompt (_) will appear.

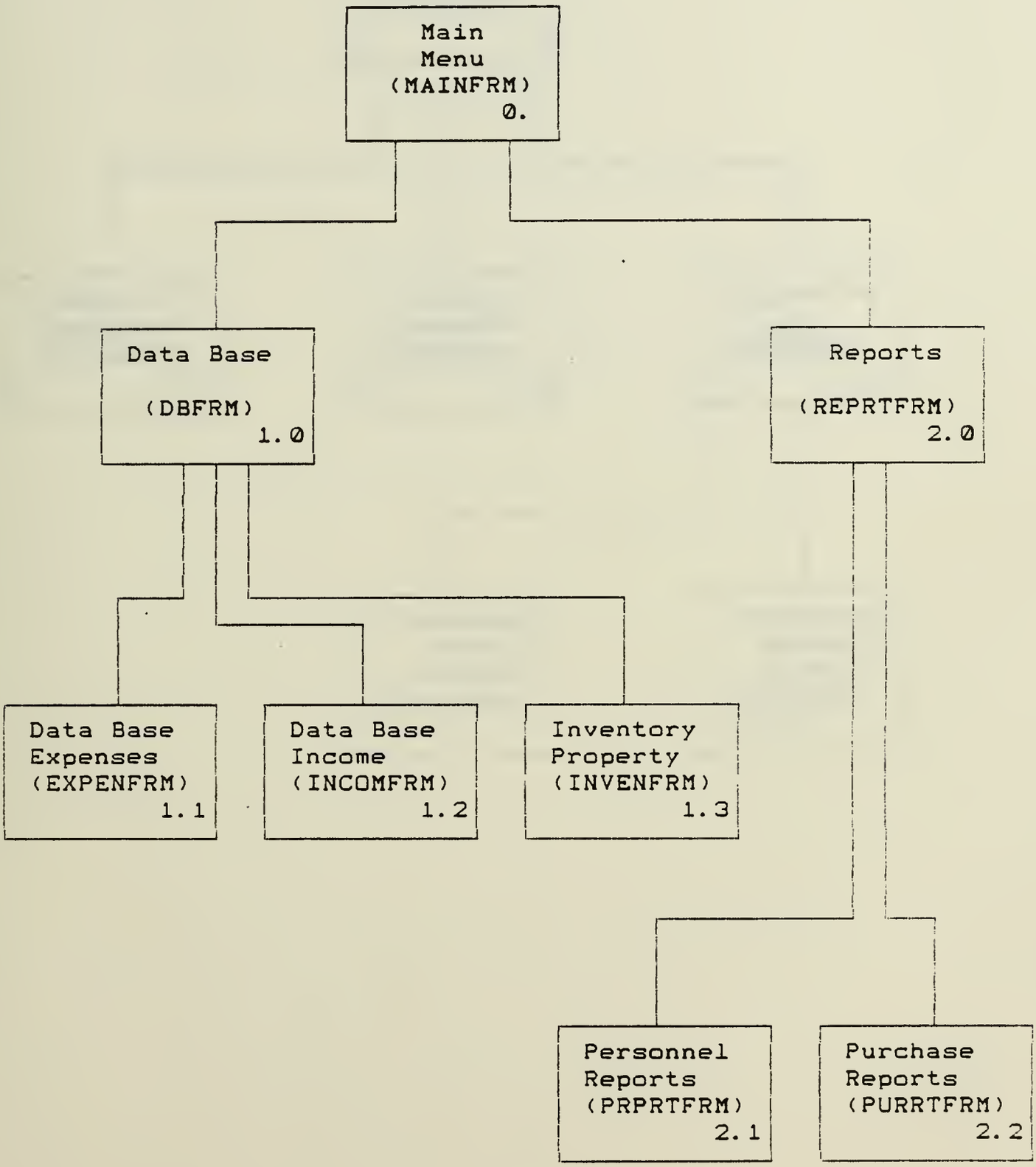
Then perform the following:

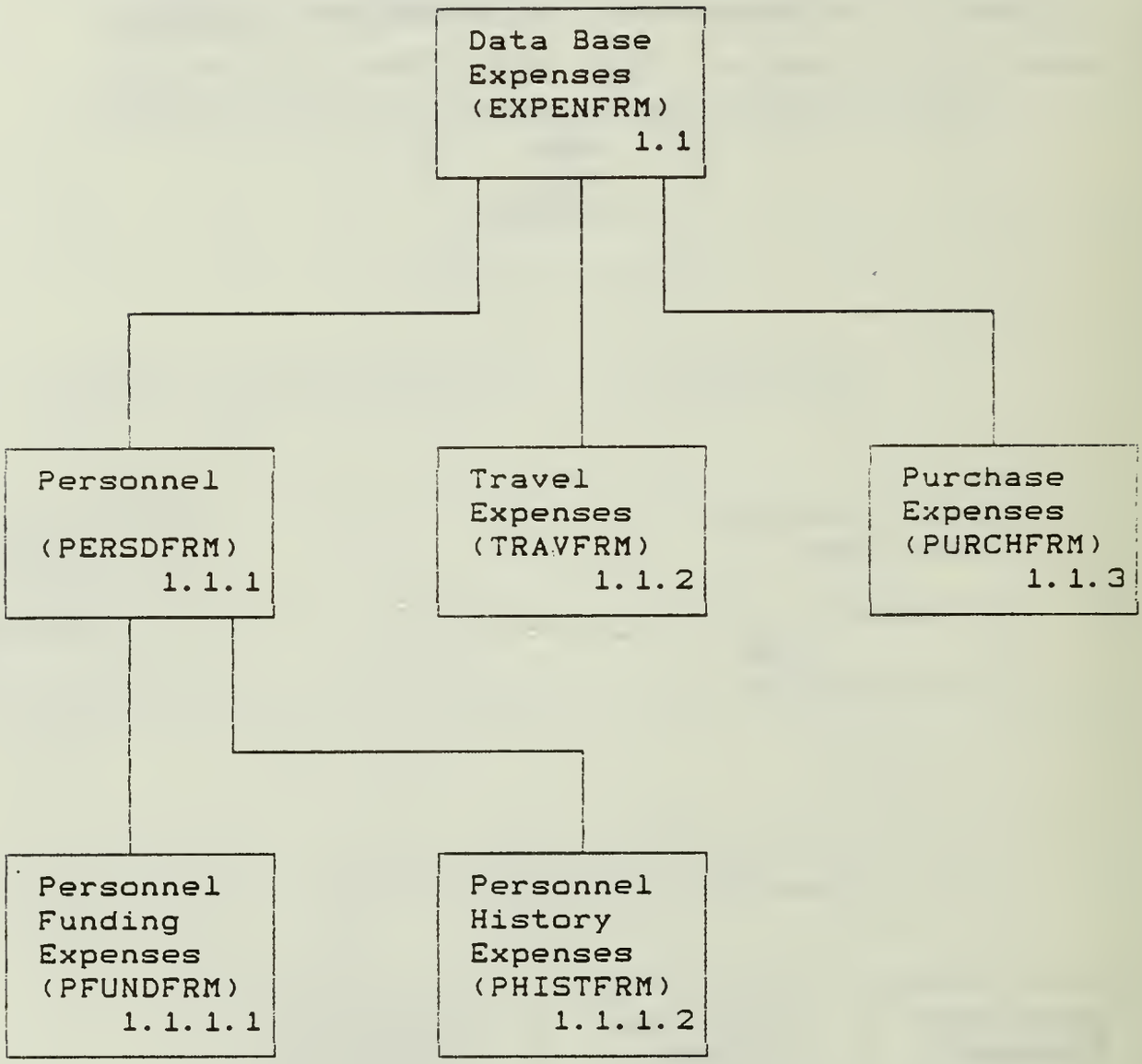
1. If processing is returned to a calling module, attempt to continue normal processing. If the problem contin-

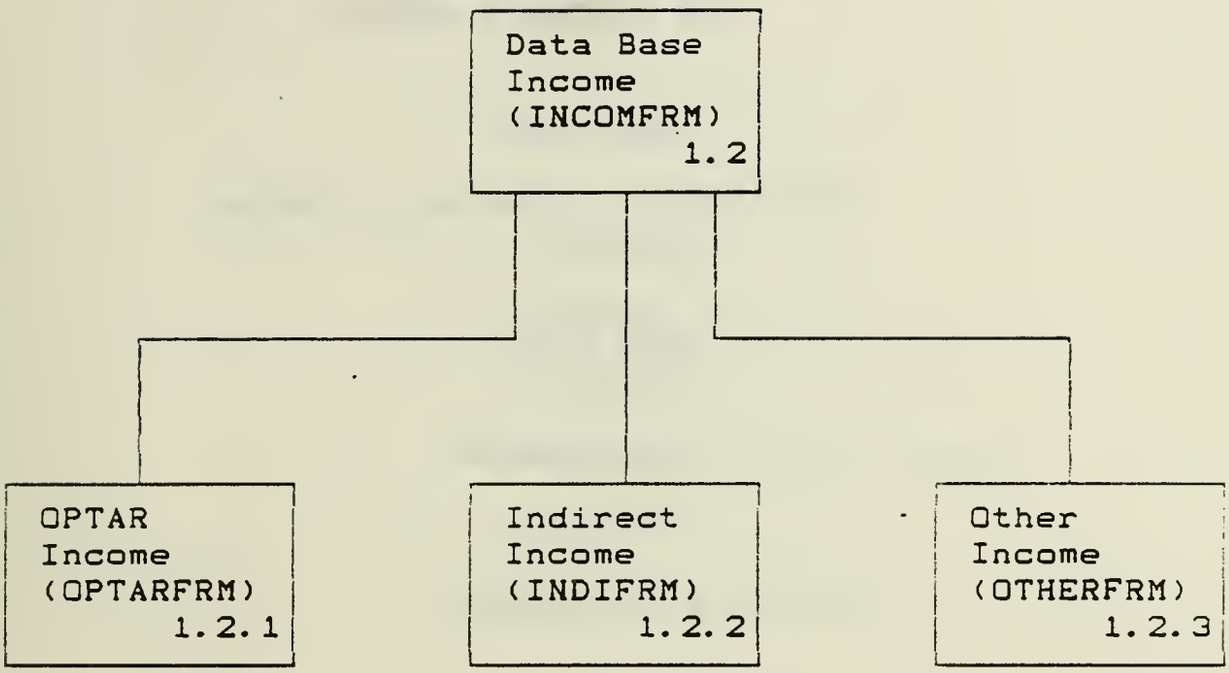
ues, turn the power off, remove the diskette and notify the System Administrator.

2. If the KnowledgeMan prompt (_) appears, type in KMAN MAINMEN and attempt to continue normal processing. If the problem continues, turn the power off, remove the diskette and notify the System Administrator.

APPENDIX B
Diagram of the AS System's Menus




Data Base
Expenses
(EXPENFRM)
1.1
Personnel
(PERSDFRM)
1.1.1
Travel
Expenses
(TRAVFRM)
1.1.2
Purchase
Expenses
(PURCHFRM)
1.1.3
Personnel
Funding
Expenses
(PFUNDFRM)
1.1.1.1
Personnel
History
Expenses
(PHISTFRM)
1.1.1.2


Data Base
Income
(INCOMFRM)
1.2
OPTAR
Income
(OPTARFRM)
1.2.1
Indirect
Income
(INDIFRM)
1.2.2
Other
Income
(OTHERFRM)
1.2.3

APPENDIX C
AS SYSTEM'S MENUS

AS Financial Reporting System

Main Menu

1 - Data Base

2 - Reports

3 - To Quit

enter your selection _

Data Base

1 - Expenses

2 - Income

3 - Property

4 - Return to calling menu

enter your selection _

1

Database Expenses

1 - Personnel

2 - Travel

3 - Purchase

4 - Return to calling menu

enter your selection _

1.1

Personnel

1 - Personnel Funding

2 - Personnel History

3 - Return to calling menu

enter your selection _

1.1.1

Personnel Funding Expenses

1 - Browse a Table

2 - Update a Record

3 - Add a Record

4 - Delete a Record

5 - Return to calling menu

enter your selection _

1.1.1.1

Personnel History Expenses

1 - Browse a Table

2 - Update a Record

3 - Add a Record

4 - Delete a Record

5 - Return to calling menu

enter your selection _

1.1.1.2

Travel Expenses

1 - Browse a Table

2 - Update a Record

3 - Add a Record

4 - Delete a Record

5 - Return to calling menu

enter your selection _

1.1.2

Purchase Expenses

1 - Browse a Table

2 - Update a Record

3 - Add a Record

4 - Delete a Record

5 - Return to calling menu

enter your selection _

1.1.3

Data Base Income

1 - OPTAR

2 - Indirect

3 - Other

4 - Return to calling menu

enter your selection _

1.2

OPTAR Income

1 - Browse a Table

2 - Update a Record

3 - Add a Record

4 - Delete a Record

5 - Return to calling menu

enter your selection _

1.2.1

Indirect Income

1 - Browse a Table

2 - Update a Record

3 - Add a Record

4 - Delete a Record

5 - Return to calling menu

enter your selection _

1.2.2

Other Income

1 - Browse a Table

2 - Update a Record

3 - Add a Record

4 - Delete a Record

5 - Return to calling menu

enter your selection _

1.2.3

Inventory Property

1 - Browse a Table

2 - Update a Record

3 - Add a Record

4 - Delete a Record

5 - Return to calling menu

enter your selection _

1.3

Reports

1 - Personnel

2 - Purchase

3 - Financial

4 - Travel - Outstanding TO(s)

5 - Return to calling menu

TURN THE PRINTER ON enter your selection _

2

Personnel Reports

1 - Personnel Funding

2 - Personnel History

3 - Return to calling menu

enter your selection _

2.1

Purchase Reports

1 - Requisition Status

2 - Quarterly Inventory

3 - Return to calling menu

enter your selection _

2.2

APPENDIX D
SAMPLE AS SYSTEM REPORTS
Outstanding TO(s)

DEPARTMENT OF ADMINISTRATIVE SCIENCES — REPORT DATE: 03/10/86
REPORT: OUTSTANDING TO(s) — REVISION DATE:

COST CODE	DOC #	REQUESTOR	CLAIM	DOV #
EGD05	6TOA2226	Coffey	- - 0	
EGD05	6TOA2227	Coffey	- - 0	
EGDOO	6ITA3361	esc	- - 0	
EGDOO	6TOA3350	Greer	85-11-12	
EGDOO	6TOA3352	Schneidewind	85-12-02	
EGDOO	6ITA3351	Suchan	85-12-13	X3559
EGDOO	6TOA3353	Mackay	85-01-29	X4744
EGDOO	6TOA3354	Landeros	86-02-11	X5185
EGDOO	6TOA3355	Hale	86-02-19	X5374
EGDOO	6TOA3361	Elam	- - 0	
EGDOO	6TOA3356	Straub	- - 0	
EGDOO	6TOA3357	Widmeyer	- - 0	
EGDOO	6TOA3358	Conger	- - 0	
EGDOO	6ITA3359	Pracht	- - 0	
EGDOO	6ITA3360	Schocken	- - 0	
GUCTN	6TOA4820	Haag	85-11-25	
GUCTN	6TOA4821	Bui	- - 0	
NONE	851201	Evered	- - 0	
NONE	851202	McMasters	- - 0	
RCAZ2	6TOA7002	Lapatra	85-11-20	
RCAZ2	6TOA7001	BUDDENBERG	85-12-04	X3168
RCAZ2	6TOA7003	LaPatra	85-12-12	X3524
RCAZ2	6TOA7005	Jones	86-01-29	X4743
RCFCF	6TOB7725	Creighton	- - 0	
RCFCF	6TOA7727	Creighton	- - 0	
RCFCF	6TOB7726	Creighton	- - 0	
RCFCF	6TOB7727	Creighton	- - 0	

Page 1

Personnel Funding

DEPARTMENT OF ADMINISTRATIVE SCIENCES
REPORT: PERSONNEL FUNDING

REPORT DATE: 03/10/86

NAME	FUND FROM	FUND TO	HRS/WK	DAYS	HOURS	COST CODE	RATE	AMOUNT
	- - 0	- - 0	0		0		$ 0.00	$ 0.00
	- - 0	- - 0	0		0		$ 0.00	$ 0.00
Allen, Monique	85-10-14	85-10-14	0	01	0	XEOLL	$ 0.00	$ 0.00
	85-10-15	85-11-10	0	19	0	RWCS1	$ 0.00	$ 1362.49
	85-11-11	85-11-11	0	01	0	XEOLL	$ 0.00	$ 0.00
	85-11-12	85-11-27	0	12	0	RWCS1	$ 0.00	$ 860.52
	85-11-28	85-11-28	0	01	0	XEOLL	$ 0.00	$ 0.00
	85-11-29	85-12-07	0	06	0	RWCS1	$ 0.00	$ 430.26
	85-12-08	85-12-24	0	12	0	RWCS1	$ 0.00	$ 860.52
	85-12-25	85-12-25	0	01	0	XEOLL	$ 0.00	$ 0.00
	85-12-26	85-12-31	0	04	0	RWCS1	$ 0.00	$ 286.84
	86-01-01	86-01-01	0	01	0	XEOLL	$ 0.00	$ 0.00
	86-01-02	86-01-05	0	02	0	RWCS1	$ 0.00	$ 143.42
	86-01-06	86-01-08	0	03	0	XALSL	$ 0.00	$ 0.00
	86-01-09	86-01-19	0	09	0	RWCS1	$ 0.00	$ 645.39
	86-01-20	86-01-20	0	01	0	XEOLL	$ 0.00	$ 0.00
	86-01-21	86-02-05	0	12	0	RWCS1	$ 0.00	$ 860.52
	86-02-17	86-02-17	0	01	0	XEOLL	$ 0.00	$ 0.00
	86-10-01	85-10-13	0	09	0	RWCS1	$ 0.00	$ 645.39
Balesteri, Angela	85-10-01	85-10-13	0	06	0	RLQZP	$ 0.00	$ 545.40
	85-10-14	85-10-14	0	01	0	XEOLL	$ 0.00	$ 0.00
	85-10-15	85-11-10	0	12	0	RLQZP	$ 0.00	$ 1090.80
	85-11-11	85-11-11	0	01	0	XEOLL	$ 0.00	$ 0.00
	85-11-12	85-11-27	0	08	0	RLQZP	$ 0.00	$ 727.20
	85-11-28	85-11-28	0	01	0	XEOLL	$ 0.00	$ 0.00
	85-11-29	85-12-22	0	10	0	RLQZP	$ 0.00	$ 909.00

Personnel History

DEPARTMENT OF ADMINISTRATIVE SCIENCES
REPORT TITLE: PERSONNEL HISTORY

REPORT DATE: 03/10/86

NAME	JOB TITLE	SERIES	GRADE	STEP	PD#	BILLET#	EXPIRE	WORK UNIT	WORK DIREC
Allen, Monique	Research Technician	303	5	1	A0656	918g	86-09-30	Func.Skill	Sticht
Balesteri, Angela	Editorial Asst. (Typ	1087	5	9	A0505	304g	86-09-23		San Miguel
Buttlee-Miller, Beth	Management Assistant	344	7	1	A0140	827g	86-08-31		Dickey
Cochrane, Candi	Research Technician	303	5	1	A1065	954g	86-09-30		Euske
Covington, Maria	Travel Clerk (Typing	2132	4	10	A0487	135g	- - 0		Dickey
Davis, Helen	Statistician (Ops. &	1530	11	1	A0836	877g	86-10-01		Thomas
Dickey, Laurence	Support Services Sup	342	9	1	A0871	881g	86-05-14		Greer
George, Karen	Technical Manuals Wr	1083	7	1	A1085	913g	86-10-10	Func.Skill	Armijo
Gouveia, Cynthia	Copier/Duplicating O	350	2	1	A1037	843g	87-01-11		Dickey
Howard, LaVerne	Clerk-Typist	322	4	4	A1078	267g	- - 0		Dickey
Humphreys, Susan	Clerk-Typist	322	4	3	A0488	134g	- - 0		Dickey
Hutchings, Renske	Research Technician	303	5	1	A0938	896g	86-06-02		Thomas
james, greta	clerk-typist	322	3	1	all44	262g	- - 0		dickey
Kocher, Kathryn	Labor Economist	110	11	7	A0496	915g	86-03-26		Thomas
Krauthammer, William	Social Services Aid	3186	4	1	A0671	864g	86-07-21	MARC	Zimmerman
Lathrop, Mary Ellen	Social Science Analy	101	7	1	A0497	862g	87-01-02		Wiepoer-Turp
Lindsay, Caryl	Computer Programmer	334	7	1	A0490	959g	86-07-01		Zoyang
Lindstrom, Jodie	Secretary (Typing)	350	5	1	A0379	893g	86-06-30		Zoyang

Quarterly Inventory

DEPARTMENT OF ADMINISTRATIVE SCIENCES
REPORT: QUARTERLY INVENTORY

REPORT DATE: 03/10/86

PA# / PO#	Cost	Vendor	Received	Issue	Description / Location	Serial#
1 / 1111	$ 1000.00	IBM	11-11-11	ADAMS	PD-JR / I-270	11111
222 / 2222	$ 2000.00	HONEYWELL	- -22	BAKER	COMPUTER / I-280	22222
333 / 333	$ 3333.00	TEST	33-33-33	CARSON	TEST DESC / I-0-1	3333

Requisition Status

DEPARTMENT OF ADMINISTRATIVE SCIENCES
REPORT: REQUISITION STATUS

REPORT DATE: 03/10/
REVISION DATE:

RDD	DESCRIPTION	DOC #	COSTCODE	STOCK #	PO#	REQUESTOR	VENDOR
- - 0							
- - 0							
- - 0						ms. sue conger	
- - 0							
- - 0						james	central point software
- - 0							
- - 0							
- - 0	clipper for ibm pc			8		lyons	montrey bay computerworks
- - 0							
- - 0	Tape Dispenser, 3"	5204-9140	RWCS1	7520-00-240-2417		Joelson	Ready Supply
- - 0	Report to Judy Morocco,Naval Education & Training	5219-9664	RWCS1				Federal Express
- - 0	Pyxel Visuals	5221-7506	RLPEE		85-M-4828		Pyxel Applicaitons
- - 0	Cable to connect IBM-PC tp Daisywriter Printer	5235-7508	RLPEE		85-Y-1587		Computerland

PAGE 1

APPENDIX E

ADMINISTRATIVE SCIENCES (AS)
DATA BASE REPORTING SYSTEM

PROGRAMMER'S MANUAL

TABLE OF CONTENTS

I. BACKGROUND

The Administrative Sciences (AS) Data Base Reporting System (the AS System) is an interactive, menu driven data base management system. This system is used to input, edit and store the U.S. Naval Postgraduate School's Administrative Sciences Department's financial data (i.e., the AS Department's OPTAR, data relative to the AS Department's indirect cost and research accounts, and the Department's equipment inventory) and to create financial reports. This manual is not a tutorial of the AS System's application language.

II. DATA BASE ADMINISTRATOR

The AS Department's Management Assistant is the AS System's Administrator and Data Base Administrator (DBA). The duties of the DBA includes defining data tables and structures, data security, program and systems modifications, system documentation and user training. All request or suggestions for changes and major problems must be referred to the System's Administrator.

III. SYSTEM DESIGN

The AS System performs data manipulation and report generation, and has been designed along these two functional paths. Appendix F, the AS System's data flow diagram details the two functional paths. Prototyping was used; first perform the data manipulation for one table and the report generation for one report, then use similar logic for the other tables and reports.

A. PRIMARY DESIGN GOALS

The following were the primary design goals for AS System:

- error detection and correction
- ease in modifying programs
- data entry editing
- ease in entering, storing and modifying data
- ease in creating reports
- ease in operating the system
- ease of adding follow-on features (i.e., spreadsheets, graphics, etc.)

In an attempt to accomplish the above goals, the majority of this System's modules deal with the interface between the user and the data, instead of computations upon the data. Due to these design goals, the System's structure chart, Appendix G, displays a great deal of higher level fan-out and minimal lower level fan-in.

B. PROCESSING LOGIC

The following is the standard processing logic of the perform files (modules):

1. Change the necessary K-Man environment variables.
2. Receive passed data from a calling module.

3. Determine if user wants to exit module. (Will be no on first entry into module.) If no - perform step 4. If yes - perform step 8.

4. Display and accept input from a menu.

5. Error check the menu input.

5a. Display and accept input from an error form and repeat step 5 (If menu entry was an error.).

6. Perform function of module or call a subordinate module.

7. Repeat step 3.

8. Ensure that the changed K-Man environmental variables are set to the default values.

9. Change the necessary variable to ensure that a blank appears at the calling menu's prompt.

IV. APPLICATION LANGUAGE

The application language used for the AS System is KnowledgeMan (K-Man), version 1.07; which is developed by Micro Data Base Systems, Inc., Lafayette, Indiana. KnowledgeMan is a database management system which is designed to operate on a microcomputer. The AS System uses the following KnowledgeMan capabilities:

- Data Management
- Screen Input/Output Management
- Printed Forms (Reports) Management
- Procedures

Future development of the AS System can accommodate the following:

- Statistical Analysis
- Spreadsheet Analysis
- Calculations
- Functions
- Graphics
- Integration with other financial packages; i.e. IFPS

V. SCREEN INPUT/OUTPUT MANAGEMENT

The AS System uses predefined forms for menus, data entry and data display. Menus and data entry/display forms prompt the user for entries. The AS System is designed for a novice user, therefore prompts and instructions are utilized instead of command entries. Each menu and data entry/display form also edits the user's input by using KnowledgeMan's "picture" capability. If a wrong entry is made, an error form is displayed that notifies the user of an error and prompts for a correct entry. Appendix B is a hierarchical chart of the AS System's menus and Appendix C contains each system menu.

KnowledgeMan's optional interactive forms painting component (K-Paint) was used to create the System's menus and entry/display forms. The K-Paint files are included with the AS System's K-Man files.

VI. FILES

The following file name extensions are used by the AS System:

- EXE -- a K-Man execute file
- ICF -- context file, which saves a former state of K-Man processing in an encrypted format so that it can be resumed later (menus and forms)
- IGU -- user/password security file (a K-Man file)
- INC -- include file (a K-Man file)
- IPF -- perform file, which holds a sequence of K-Man commands (except for forms and menus, the code for these files is contained in Appendix J)
- OVL -- K-Man overlay files
- TPL -- report templates (created with K-Report)

A. PERFORM FILES

The following are the AS System's perform files:

brwtable
crtrecrd
dbmen
deltrec1
deltrec2
expenmen
incommen
indirdlt
indirmen
indirupd
invendlt
invenmen
invenrpt
invenupd
outstrpt
persdmen
pfunddlt
pfundmen
pfundrpt
pfundupd
phistdlt
phistmen
phistrpt
phistupd
prprtmen
purchdlt
purchmen
purchupd

mainmen	purrtmen
optardlt	reprtmen
optarmen	requsrpt
optarupd	travlmen
otherdlt	tvlexdlt
othermen	tvlexupd
otherupd	updrecrd

Naming convention:

- *.ipf -- (* = function performed)
- *men.ipf -- (This file displays a menu and perform a selected option. * = function selected or table requested)
- *dlt.ipf -- (This file is required to delete a record from a table. * = table's name)
- *rpt.ipf -- (This file creates a report. * = report's name)
- *upd.ipf -- (This file is required to update a record in a table. * = table's name)

The above files were created with a text editor. Listings of the perform files are contained in Appendix I.

B. ERROR AND MESSAGE FORMS

The following are the AS System's error and message forms:

- delerfrm(.icf & .ipf)
- erlindir(.icf & .ipf)
- er2indir(.icf & .ipf)
- eroinven(.icf & .ipf)
- erooptar(.icf & .ipf)
- eropfund(.icf & .ipf)
- erophist(.icf & .ipf)
- erotvlex(.icf & .ipf)
- errorfrm(.icf & .ipf)

Naming convention:

- ero*.(.icf or .ipf) -- (* = table's name)

The above files were created, and may be viewed, by using K-Paint.

C. FORMS

The following are the AS System's forms:

deletfrm	tabldfrm
delqsfrm	trpldfrm
indelfrm	pfdelfrm
indircrt	pfundcrt
indirdel	pfunddel
indirect	pfunding
inupdfrm	pfupdfrm
invencrt	phdelfrm
invendel	phistcrt
inventor	phistdel
ivdelfrm	phistory
ivupdfrm	phupdfrm
opdelfrm	pudelfrm
optar	purchase
optarcrt	purchcrt
optardel	purchdel
opupdfrm	puupdfrm
otdelfrm	tvdelfrm
other	tvlexcrt
othercrt	tvlexdel
otherdel	tvlexp
otupdfrm	tvupdfrm

Naming convention:
All of the above files have an .icf or .ipf extension. The ".icf" files are saved for K-Paint usage. The ".ipf" files are performed by an AS System module (file).

- *crt. -- (A form on which a user enters data to create a record -- an entry form. * = table's name)
- *delfrm. -- (A form on which a user enters a field's value that is used to select which record may be deleted. * = table's name)
- *del. -- (A form that display's a record that may be deleted -- a deletion form. * = table's name)
- *.(icf & ipf) -- (A form that displays a record when a user elects to browse a table or update a record -- an entry form. * = table's name)
- *updfrm. -- (A form on which a user enters a field's value that is used to select which records may be updated. * = table's name)

The above files were created by using K-Paint.

D. REPORTS

The following are the AS System's reports:

- invenrpt.tpl -- Quarterly Inventory
- outstrpt.tpl -- Outstanding TO(s)
- pfundrpt.tpl -- Personnel Funding
- phistrpt.tpl -- Personnel History
- requsrpt.tpl -- Requisition Status

The above files were created, and the structure may be viewed, by using K-Report.

E. MENUS

The following are the AS Systems menus:

expenfrm	pfundfrm
incomfrm	phistfrm
indirfrm	prprtfrm
invenfrm	purchfrm
mainfrm	purrtfrm
optarfrm	reprtfrm
otherfrm	travfrm
persdfrm	

Naming convention:

- *frm.(icf & ipf) -- (* = the subject table or purpose of menu; i.e. phistfrm is the menu for the phistory table)

The above files were created by using K-Paint.

F. TABLES

The following are the tables for the AS System:

- indirect.itb -- Indirect Income
- inventor.itb -- Inventory Property
- optar.itb -- OPTAR Income
- other.itb -- Other Income
- pfunding.itb -- Personnel Funding Expenses
- phistory.itb -- Personnel History Expenses
- purchase.itb -- Purchase Expenses
- tvlexp.itb -- Travel Expenses

The above tables were created by using K-Man's command language.

VII. DATA MANAGEMENT

The AS System's data is organized into tables. Each field within each table has a specific size and type (numeric, string or logical). Each file has a "picture", which KnowledgeMan uses to edit a field's value. The formats of the AS System's tables are contained in Appendix J and the field definitions are contained in Appendix H. Because the AS System's users had previously used KnowledgeMan in an ad hoc query mode, and to decrease processing time, this is not a relational data base; there is a duplication of data between some tables.

User access may be limited to tables and fields within tables. The use of this data security capability is the responsibility of the DBA.

VII. AS SYSTEM REQUIREMENTS

A. REFERENCES

The following references are necessary for successfully programming the AS System:

- KnowledgeMan Reference Manual
- Interactive Forms Painting (K-Paint); Supplement #1 to the KnowledgeMan Reference Manual
- Full-Screen Text Processor (K-Text); Supplement #2 to the KnowledgeMan Reference Manual or another text editor
- Custom Report Generation (K-Report); Supplement #5 to the KnowledgeMan Reference Manual

The following references are helpful for successfully programming the AS System:

- Disk Operating System (version 2.1) by Microsoft Corp.
- The Beginner's Guide to KnowledgeMan
- Discovering KnowledgeMan; published by the Howard Sams Co., Indianapolis
- The Book of KnowledgeMan; by Gilbert M. Roeder

B. SOFTWARE

The following software is required for operating the AS System:

- DOS, version 2.1 -- The CONFIG.SYS file must have buffers set to at least 10. The AS System has buffers=11. Each additional buffer cost an additional 528 bytes of memory, but may result in faster processing.

Note -- A diskette has been provided to the DBA which contains DOS version 2.1 and the following commands in AUTOEXEC.BAT:

```
C:
SETCLOCK
PATH C:\AS\KMAN
CD C:\AS\SYSTEM
```

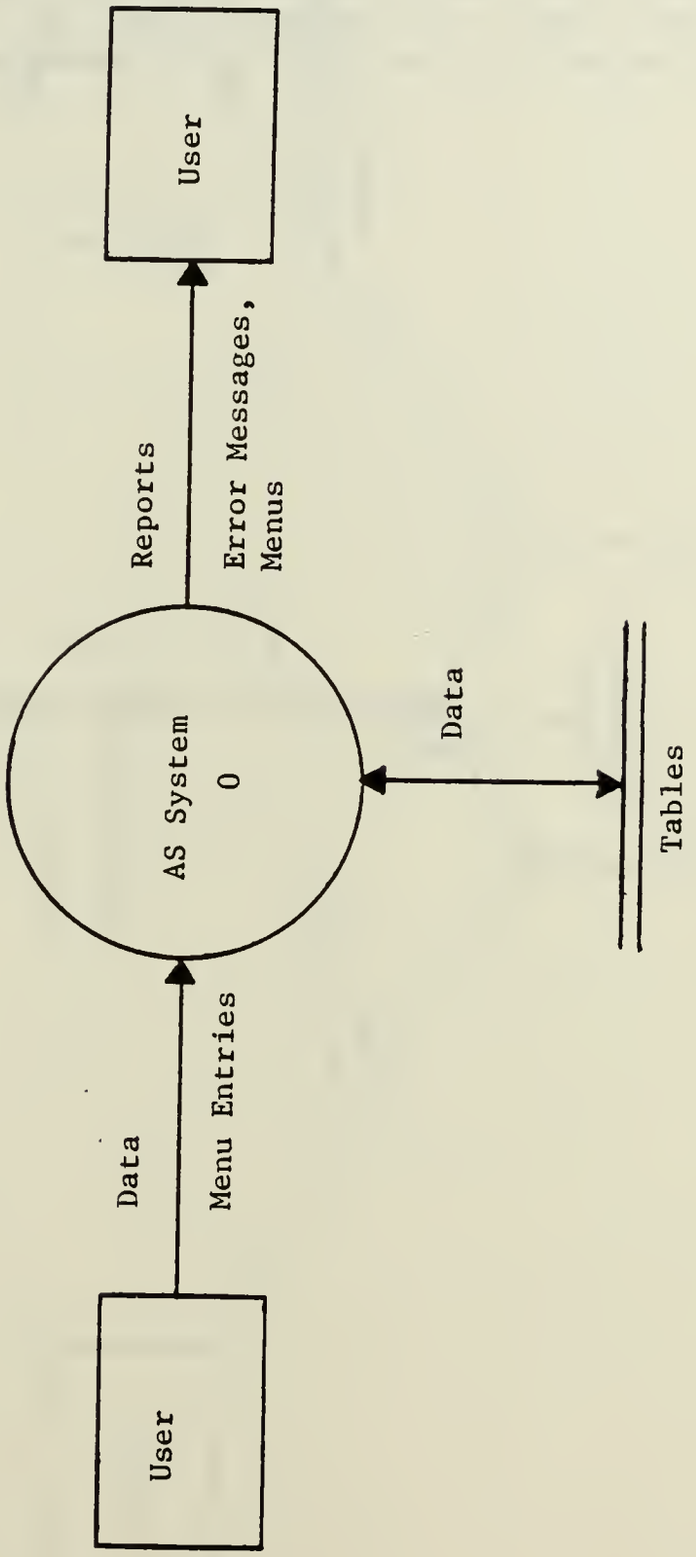
(With the current equipment, the AS System's files are stored in drive "C", sub-directory \AS\SYSTEM. KnowledgeMan, K-Paint, K-Report and K-Text are stored in "C" drive, sub-directory \AS\KMAN.)

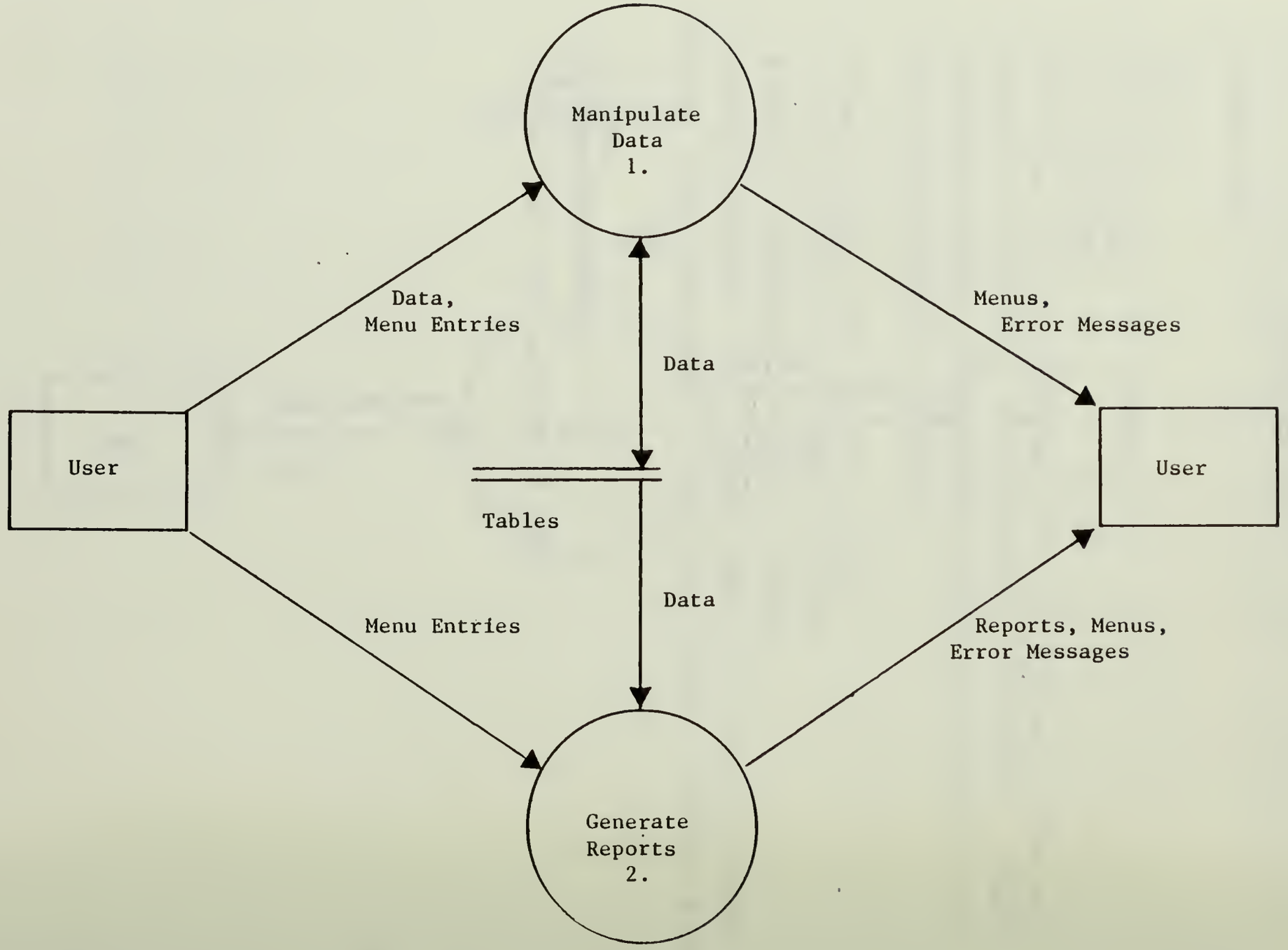
- K-Man, version 1.07 (PCDOS, IBM PC)
- K-Paint, version 1.07 (PCDOS, IBM PC)
- K-Report, version 1.07 (PCDOS, IBM PC)
- K-Text, version 1.07 (PCDOS, IBM PC) or another text editor (Kedit was used to create this system's perform files

C. HARDWARE

- Minimum RAM of 192K
- Minimum mass disk storage of 500K

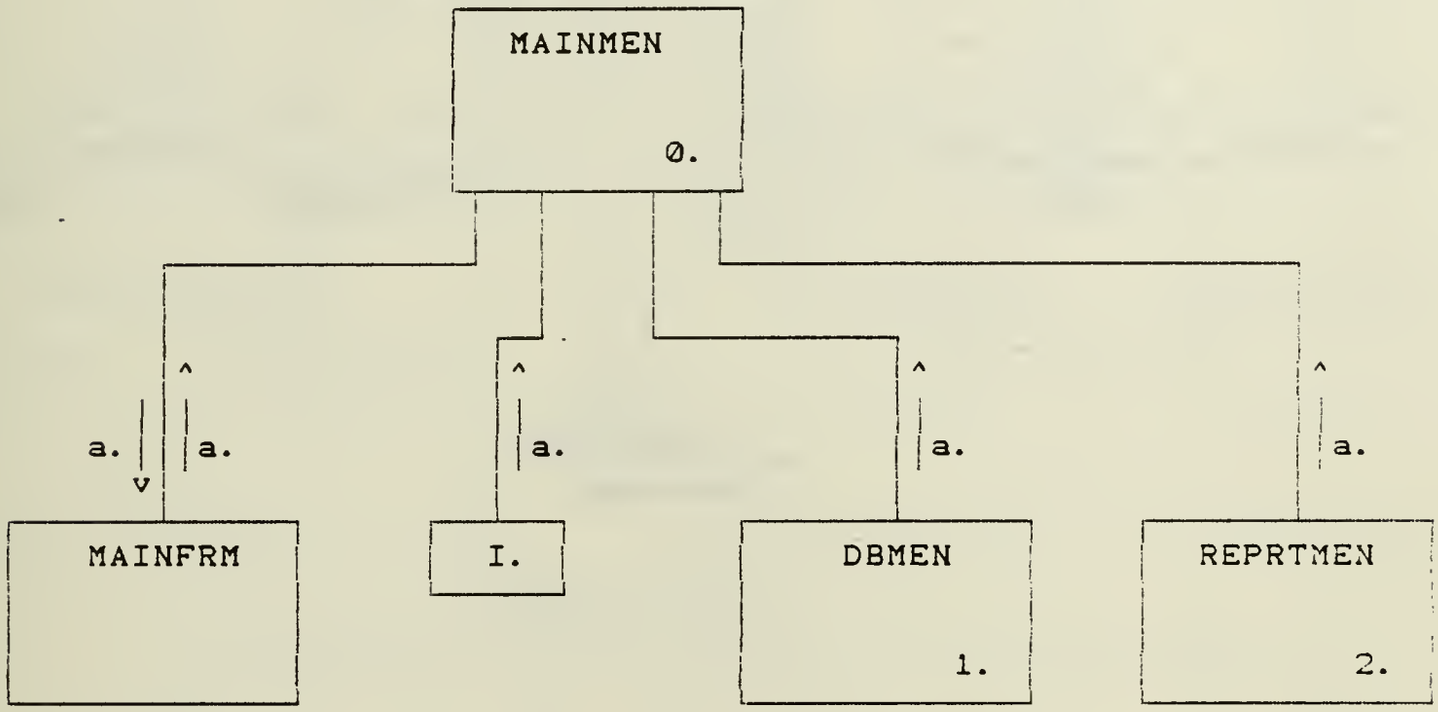
APPENDIX F
DATA FLOW DIAGRAM



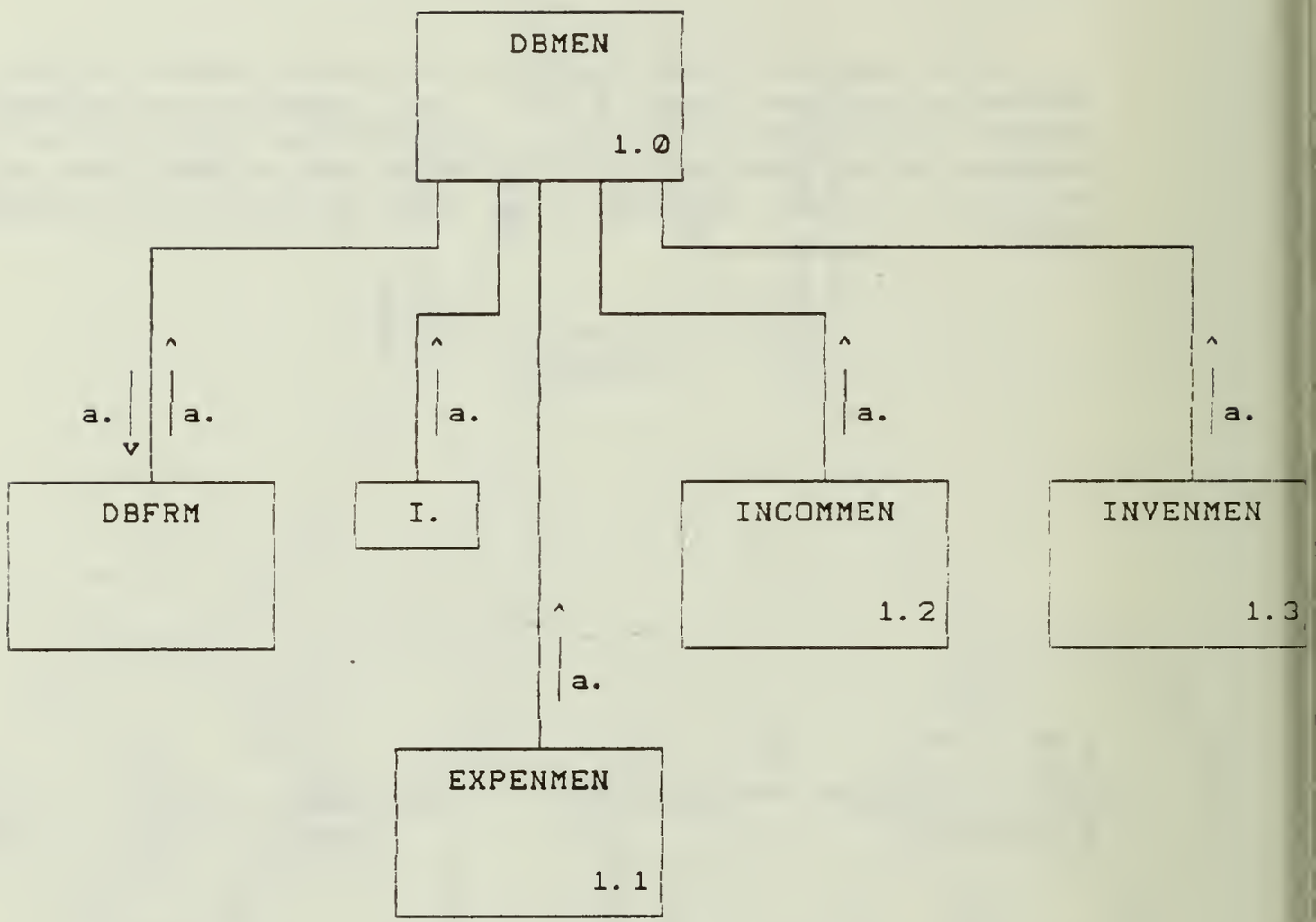

Manipulate
Data
1.
Data,
Menu Entries
Data
Menus,
Error Messages
User
Tables
User
Data
Menu Entries
Reports, Menus,
Error Messages
Generate
Reports
2.

APPENDIX G
STRUCTURE CHART

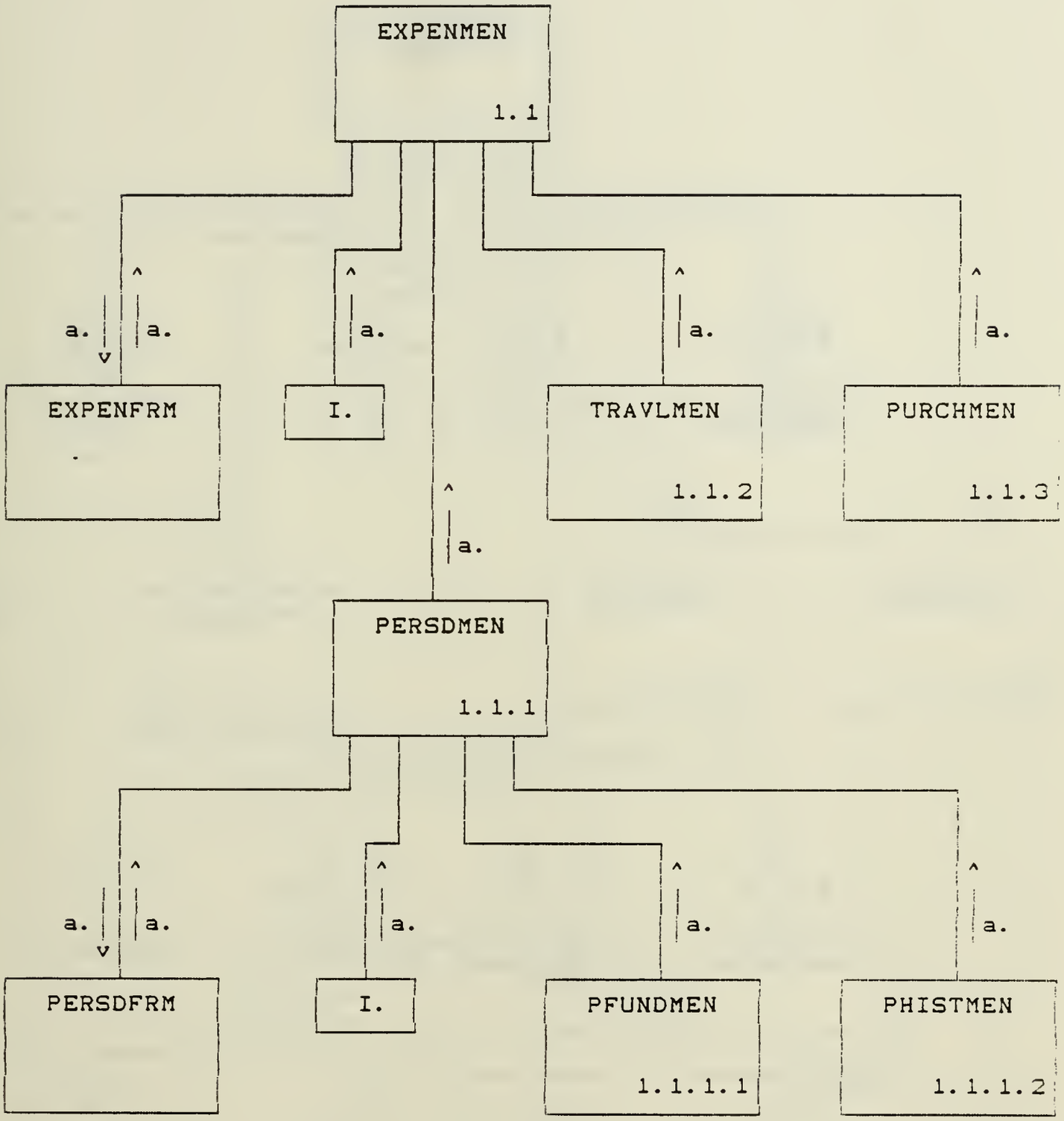
This appendix contains the structure chart of the AS System's perform files (.IPF). All modules are numbered, except the menus and forms. The modules that are called by several other modules are connected via a small box which contains a roman numeral. All modules are called conditionally.



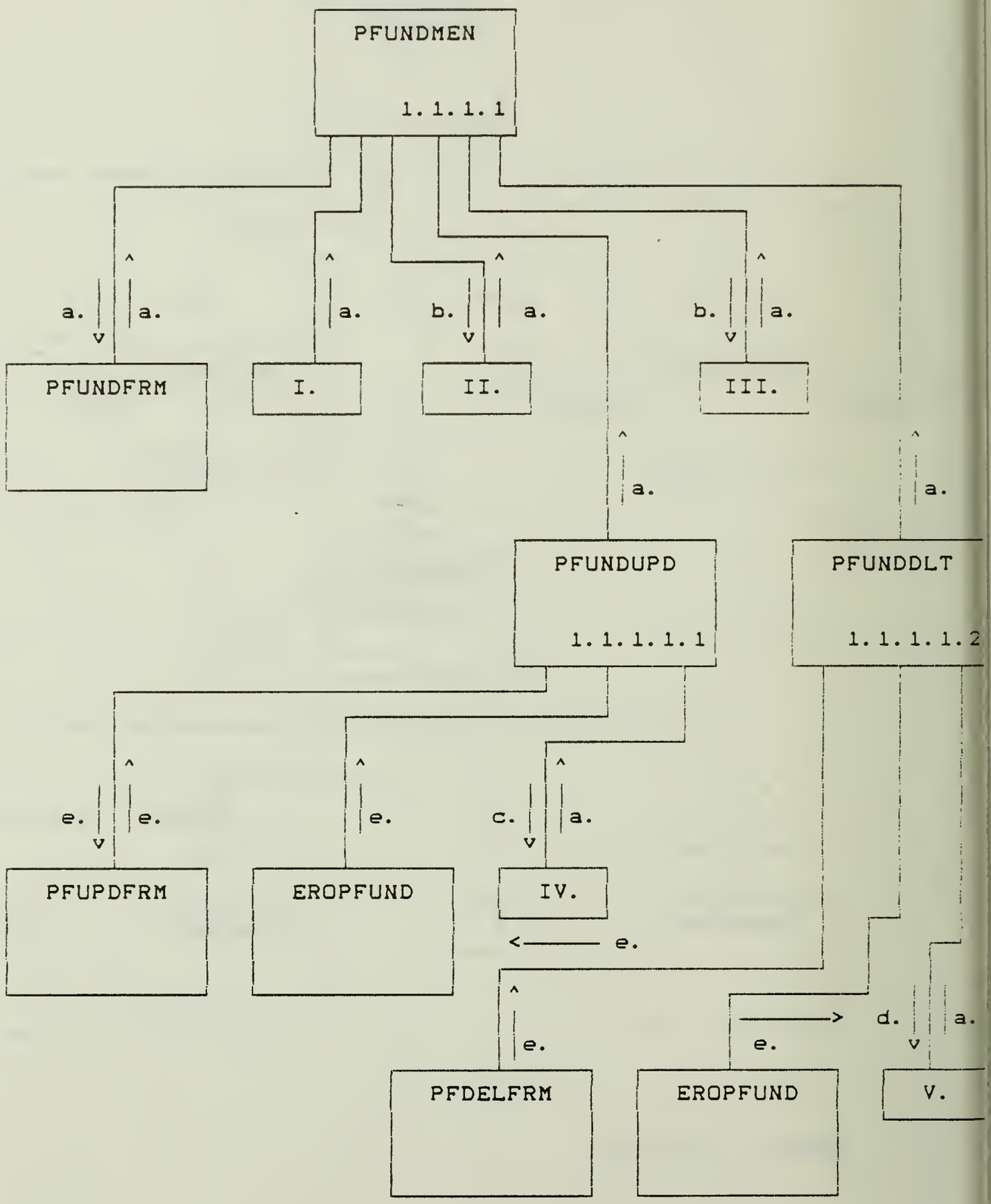
DATA: a. FORMANS



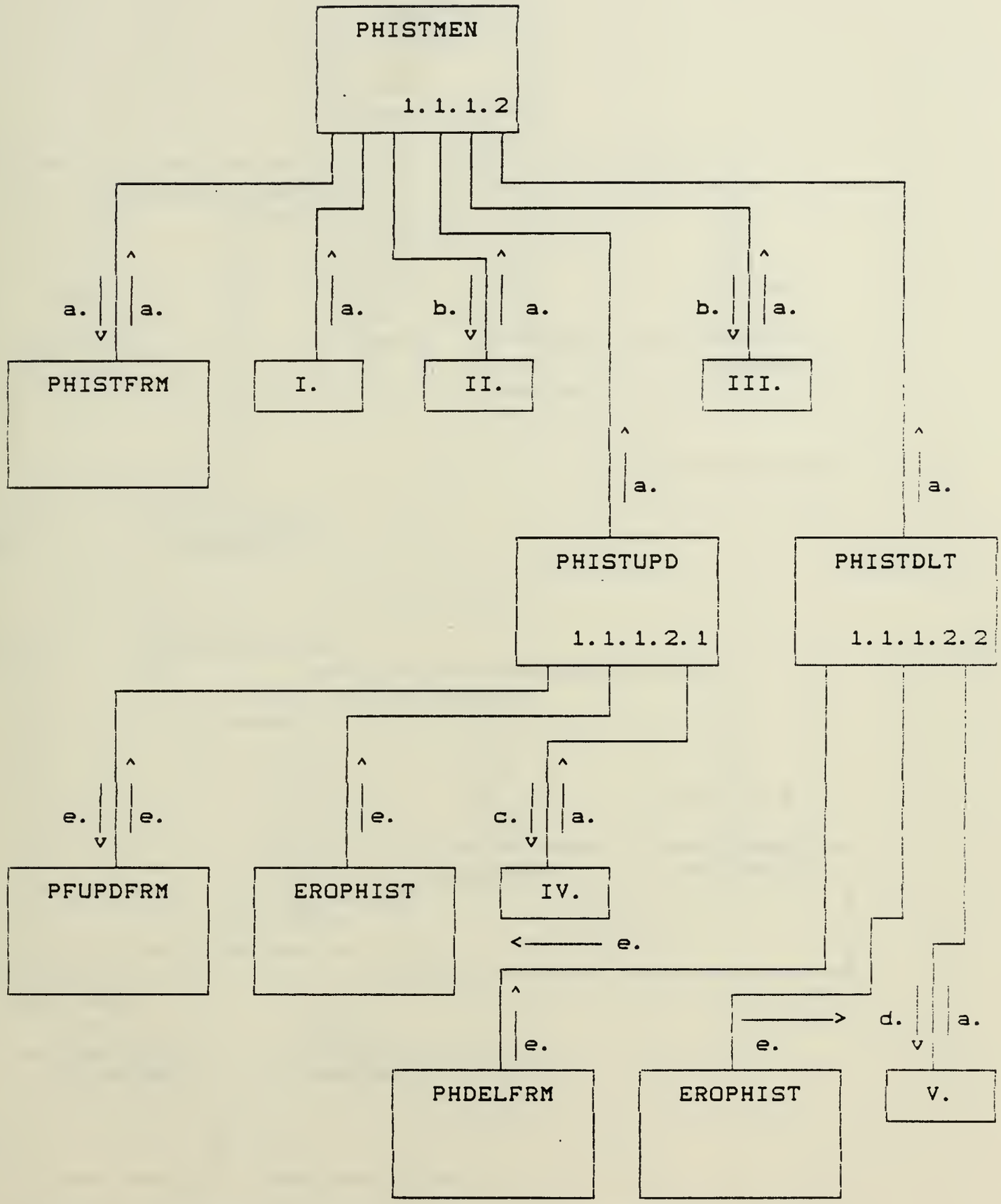
DATA: a. FORMANS



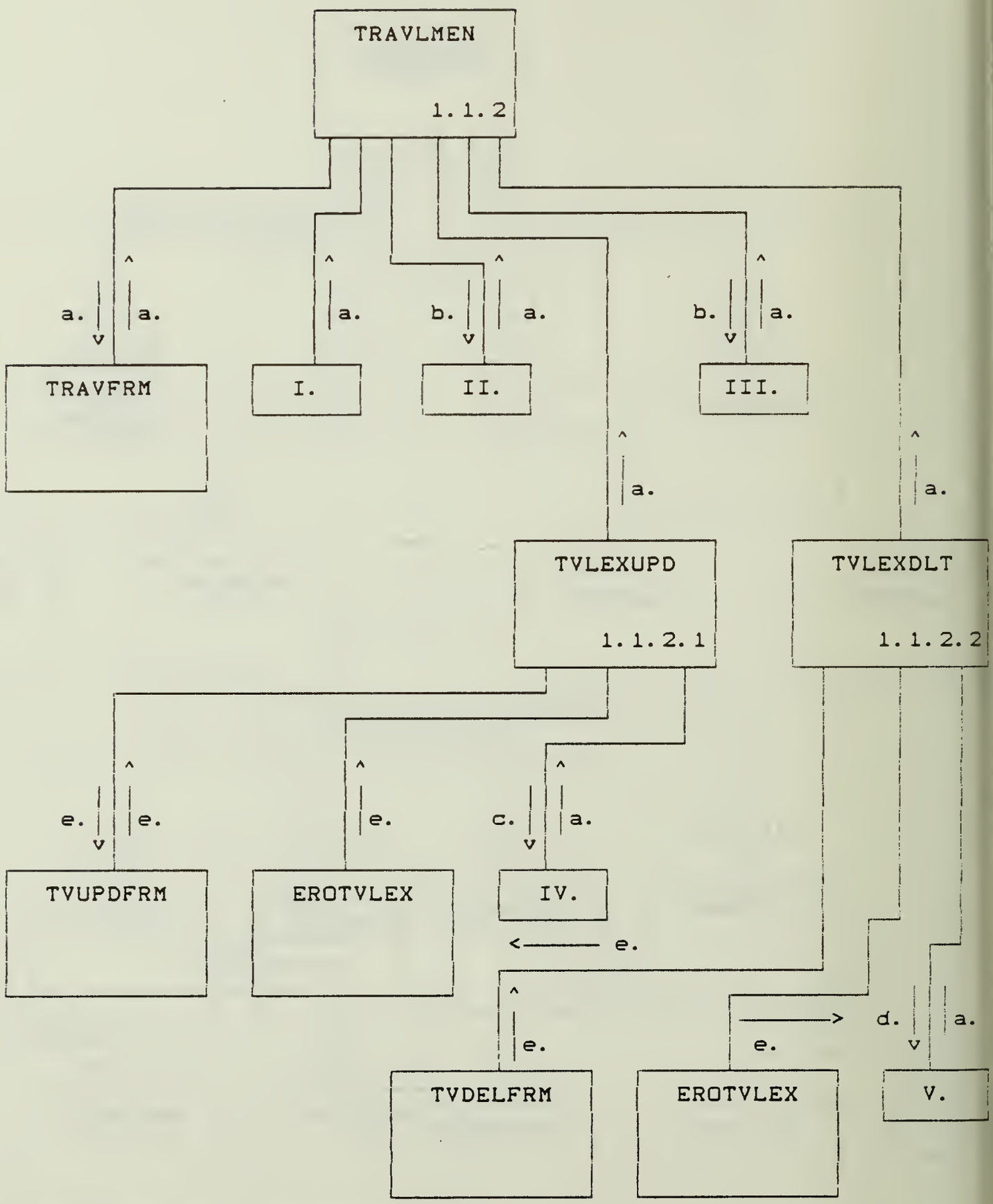
DATA: a. FORMANS



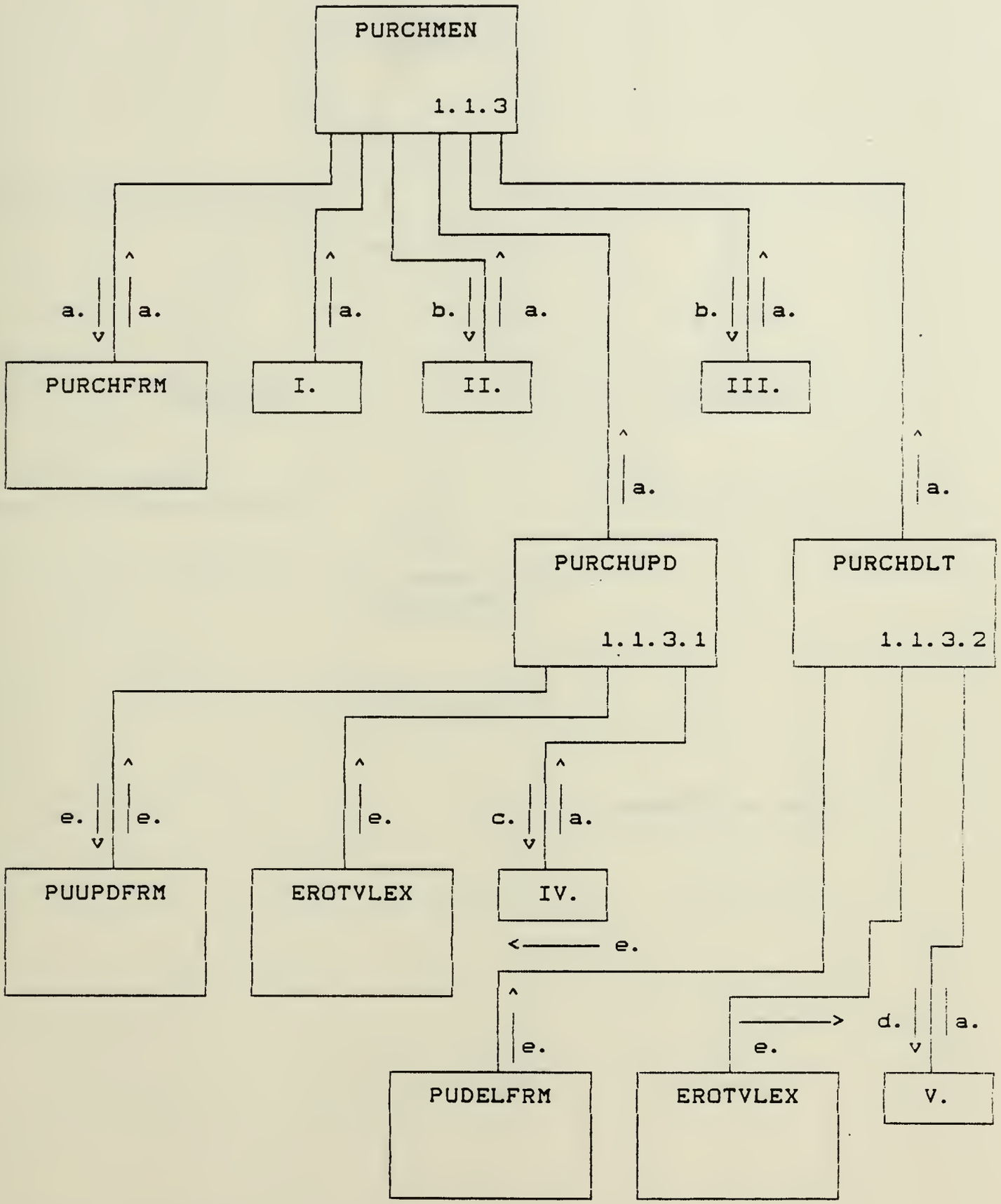
DATA: a. FORMANS; b. TABLE; c. TABLE, UPDATVAR, COND;
d. TABLE, PASSVAR, PASSCOND; e. ENTRYONE, ENTRYTWO



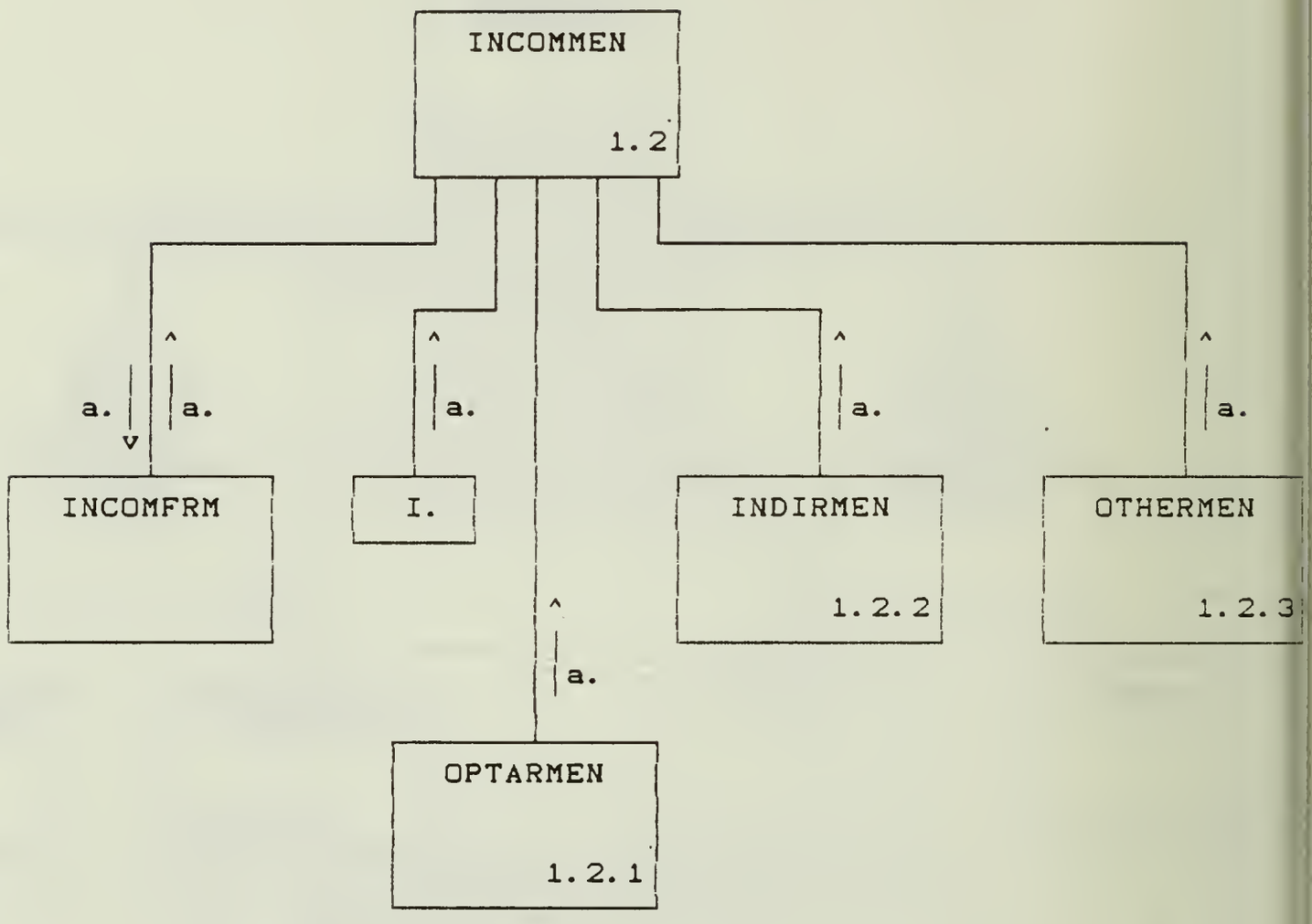
DATA: a. FORMANS; b. TABLE; c. TABLE, UPDATVAR, COND;
d. TABLE, PASSVAR, PASSCOND; e. ENTRYONE



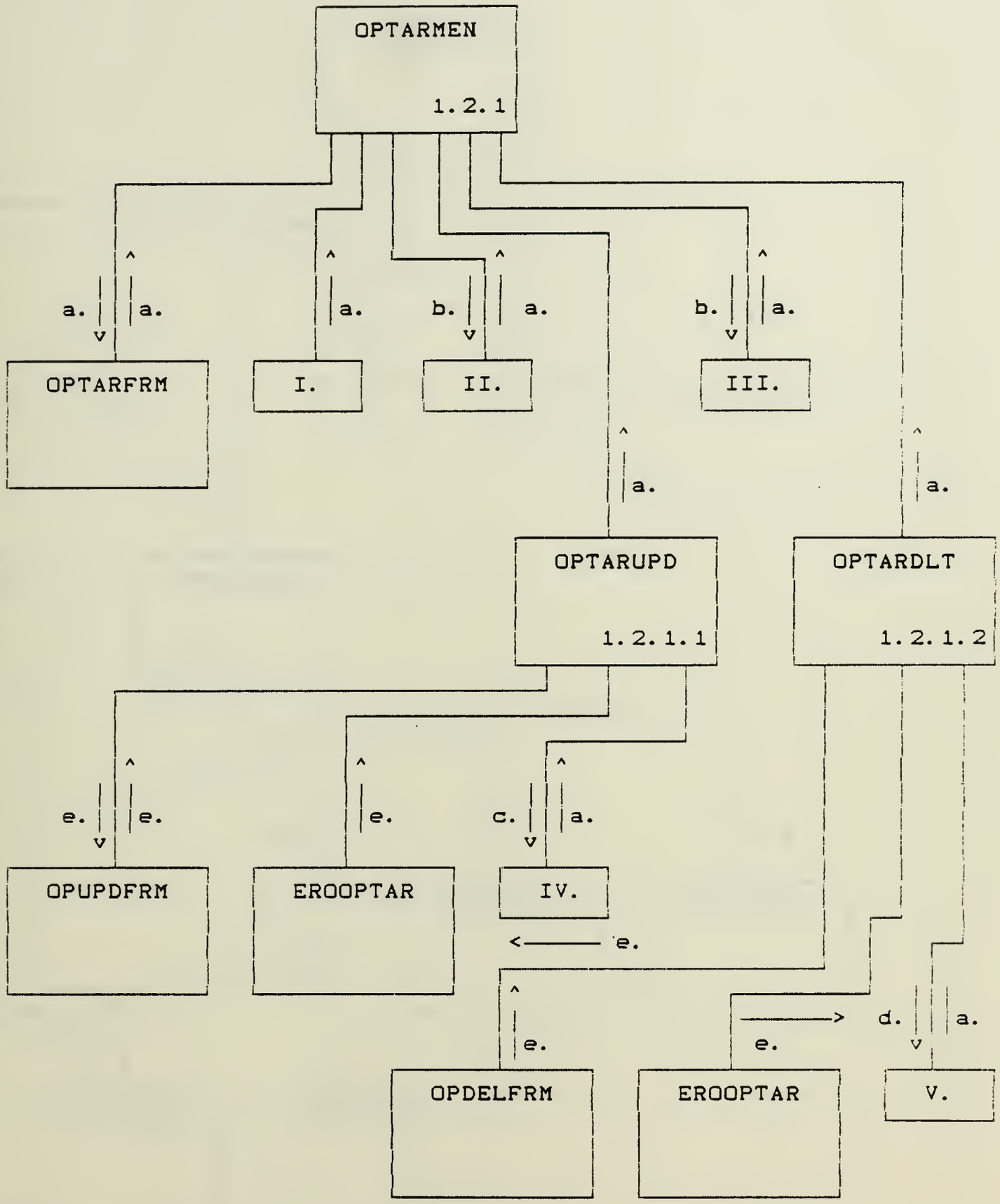
DATA: a. FORMANS; b. TABLE; c. TABLE, UPDATVAR, COND;
d. TABLE, PASSVAR, PASSCOND; e. ENTRYONE



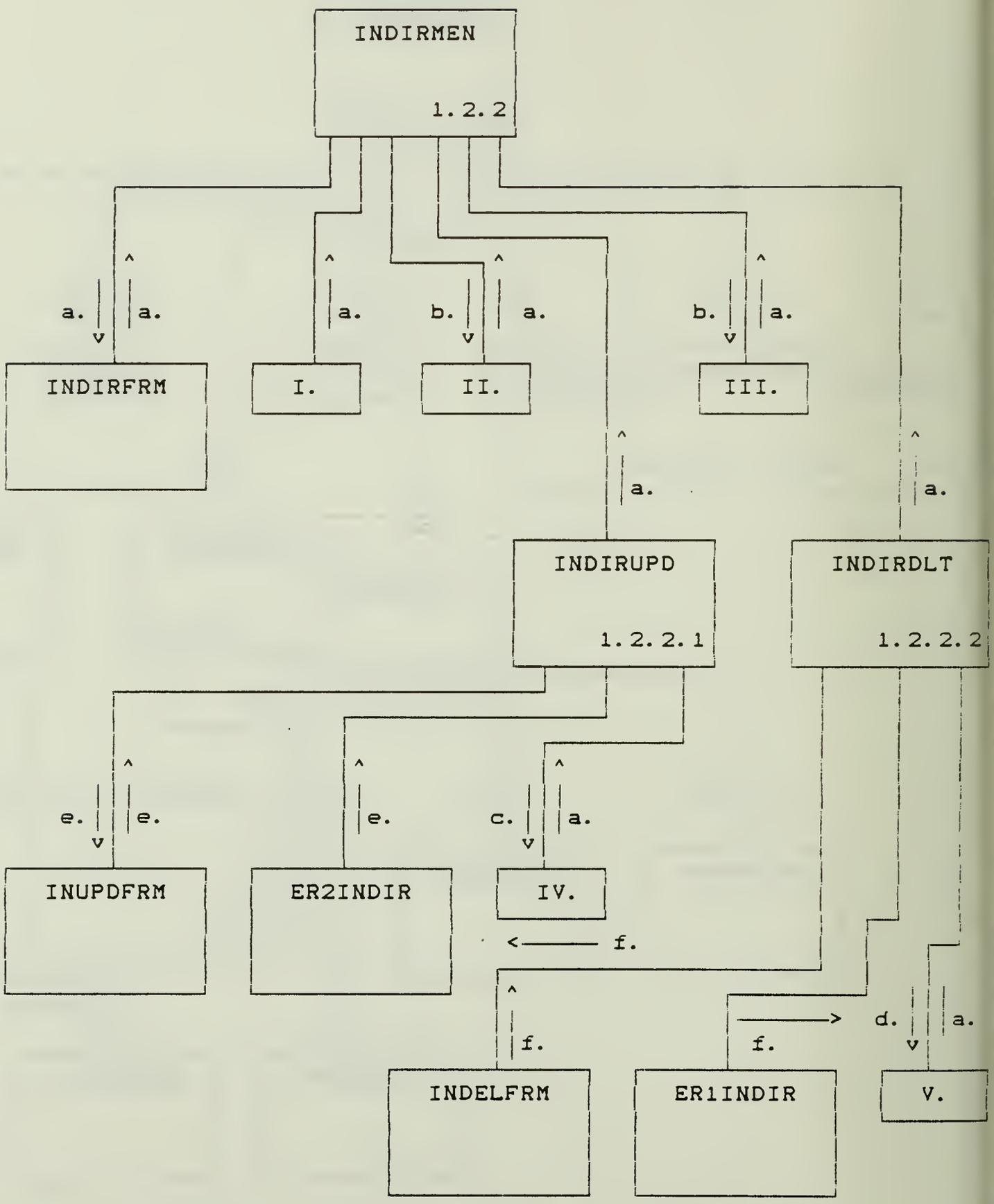
DATA: a. FORMANS; b. TABLE; c. TABLE, UPDATVAR, COND;
d. TABLE, PASSVAR, PASSCOND; e. ENTRYONE



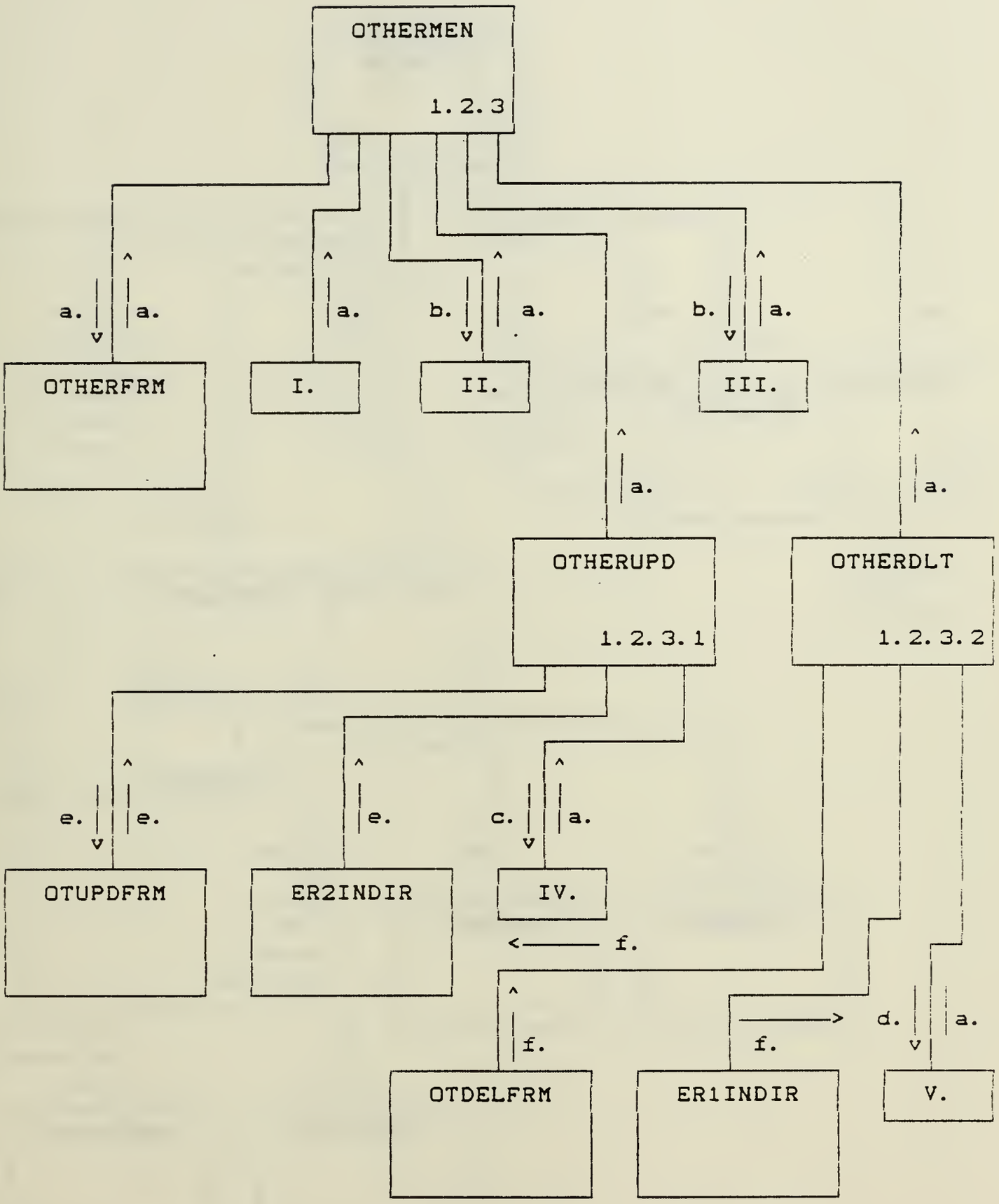
DATA: a. FORMANS



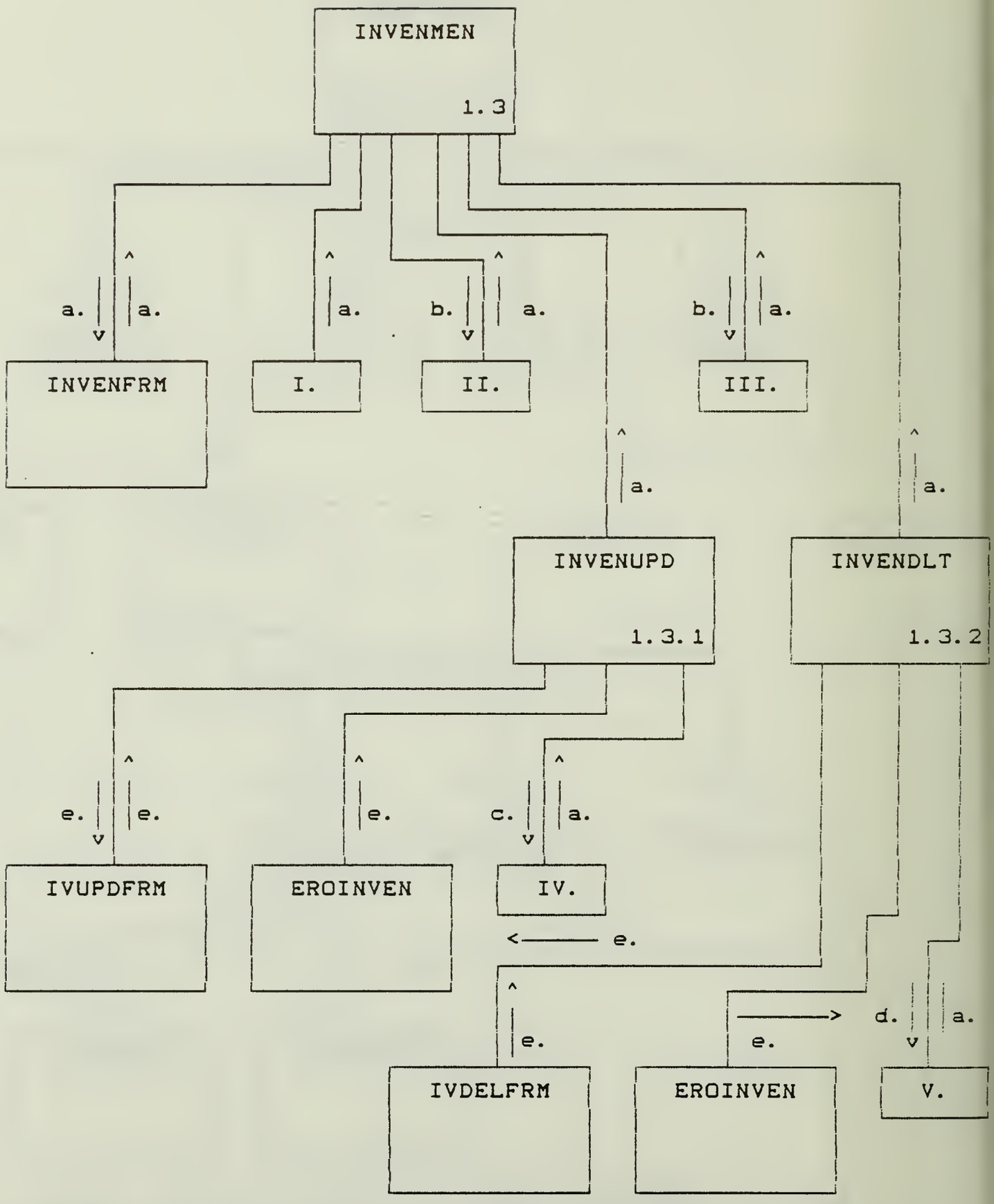
DATA: a. FORMANS; b. TABLE; c. TABLE, UPDATVAR, COND;
d. TABLE, PASSVAR, PASSCOND; e. ENTRYONE, ENTRYTWO



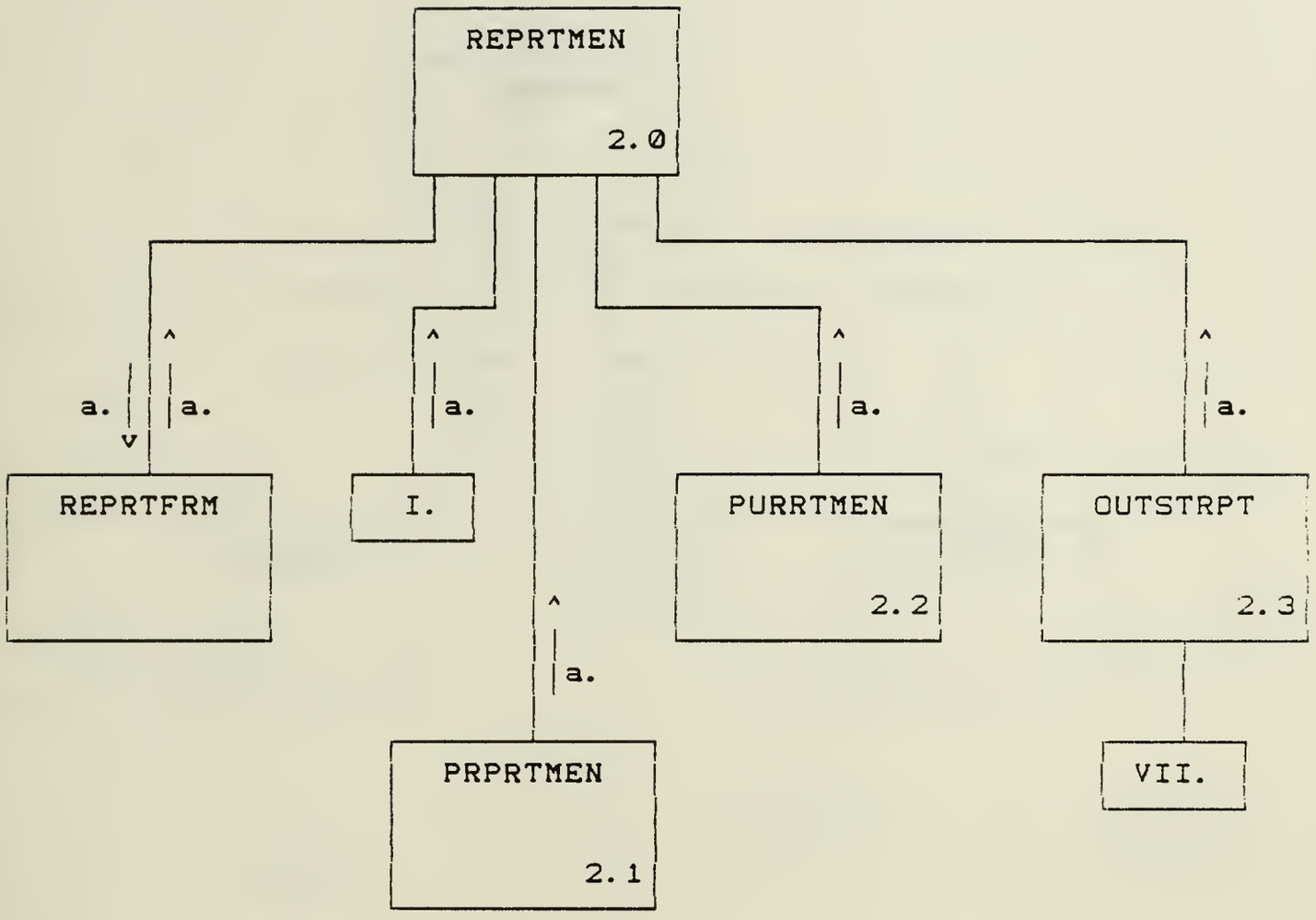
DATA: a. FORMANS; b. TABLE; c. TABLE, UPDATVAR, COND; d. TABLE, PASSVAR, PASSCOND; e. ENTRYONE, ENTRYTWO; f. ENTRYONE



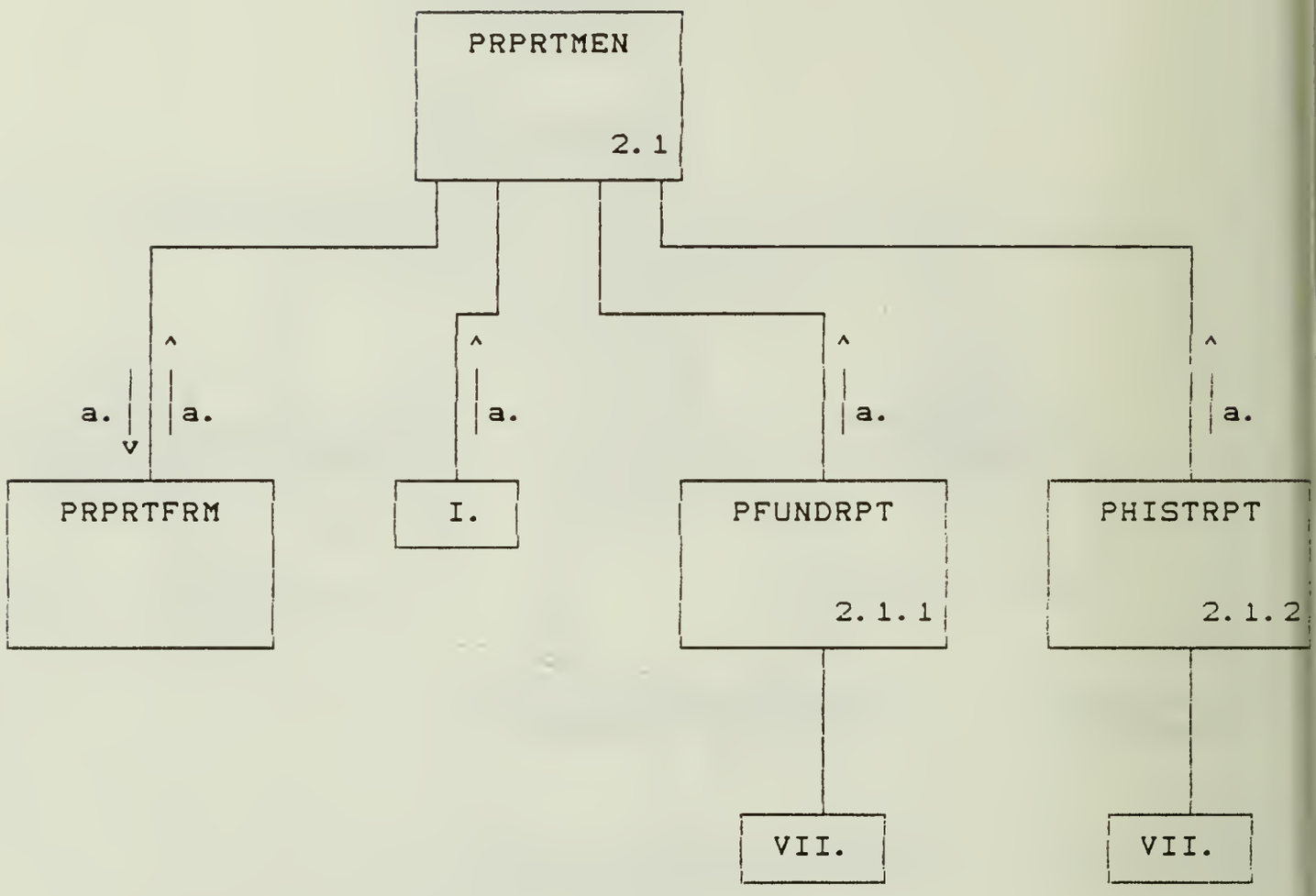
DATA: a. FORMANS; b. TABLE; c. TABLE, UPDATVAR, COND; d. TABLE, PASSVAR, PASSCOND; e. ENTRYONE, ENTRYTWO; f. ENTRYONE



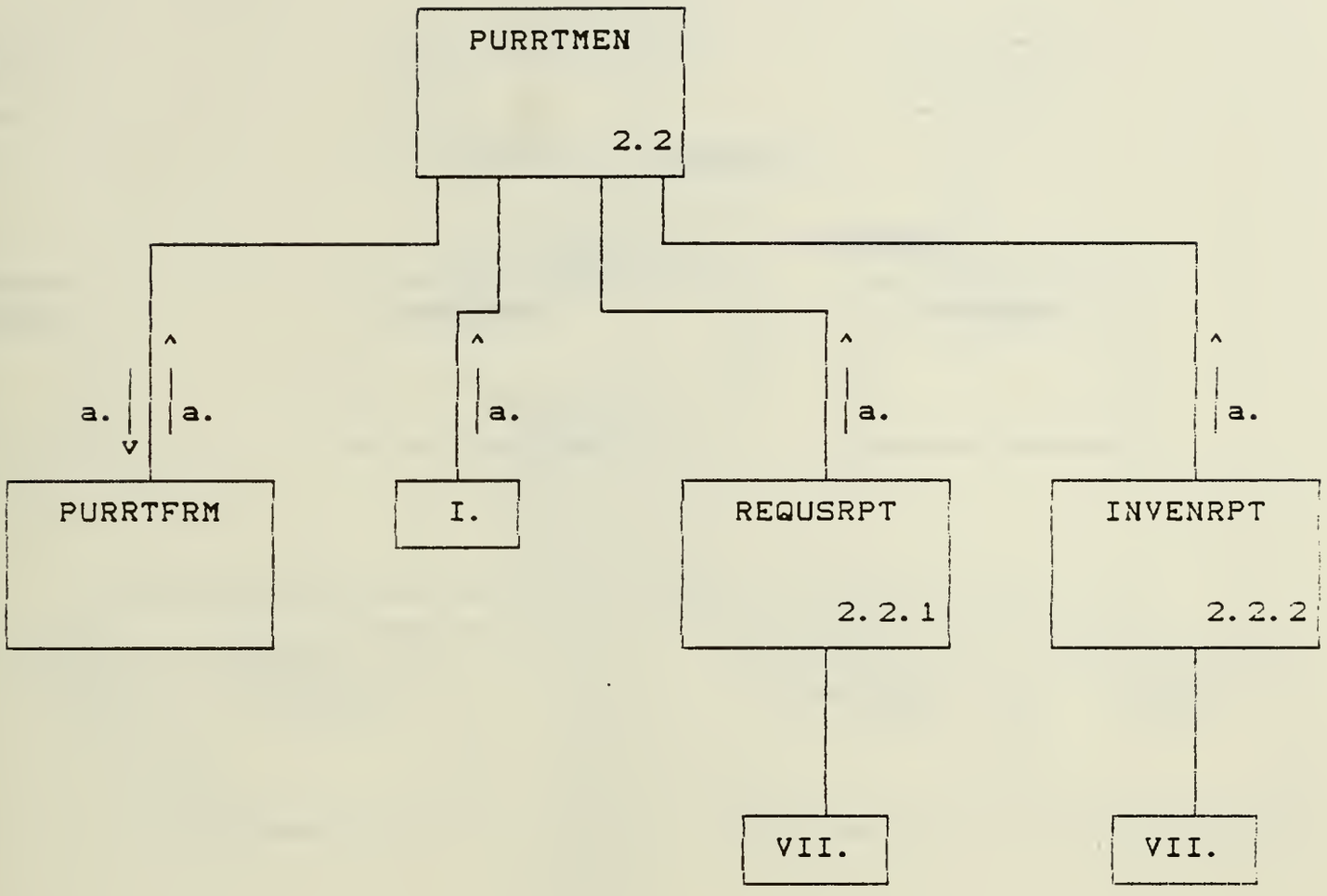
DATA: a. FORMANS; b. TABLE; c. TABLE, UPDATVAR, COND;
d. TABLE, PASSVAR, PASSCOND; e. ENTRYONE, ENTRYTWO



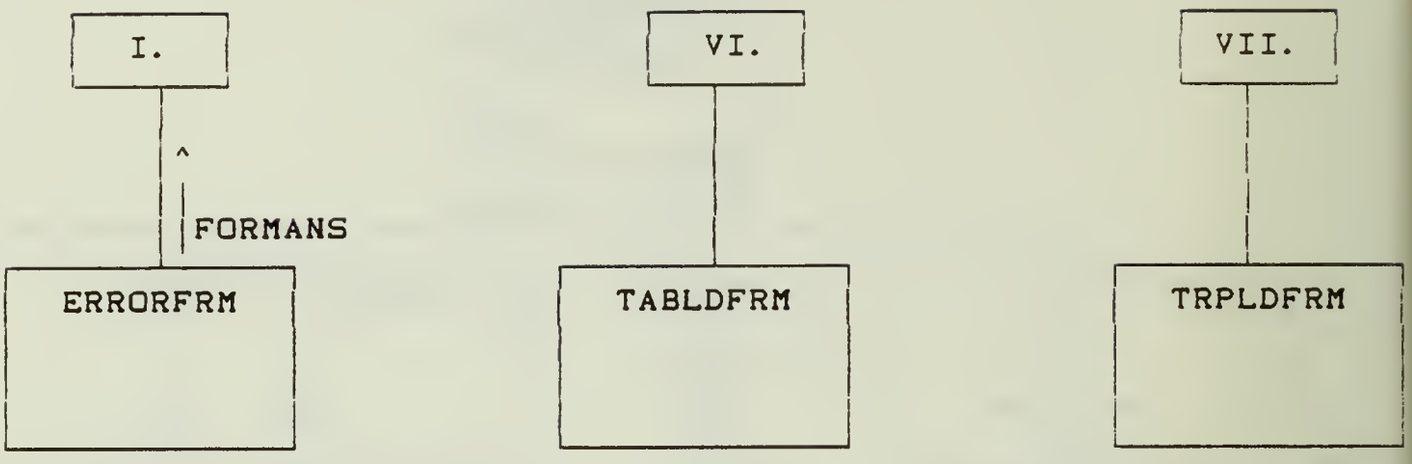
DATA: a. FORMANS

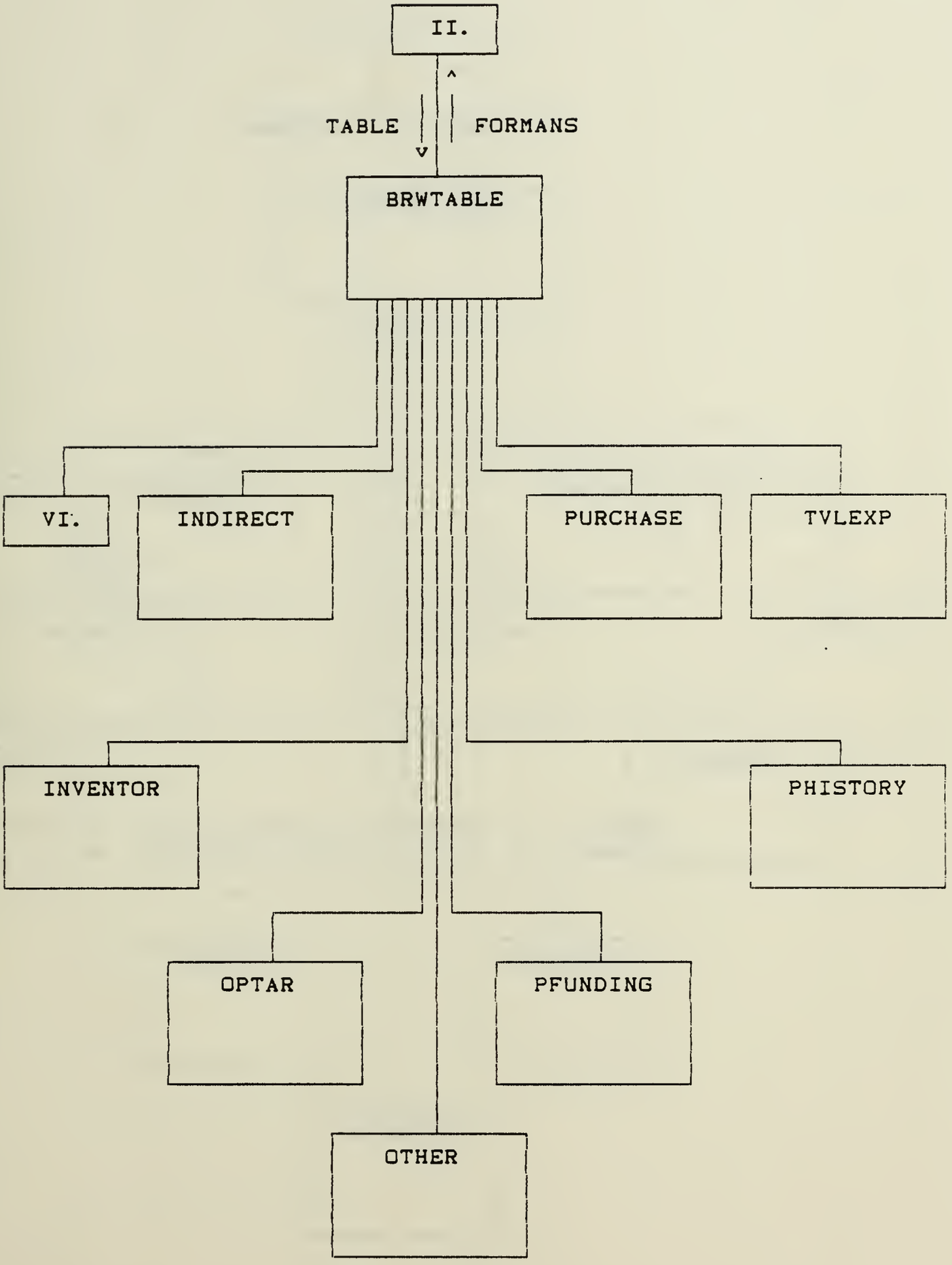


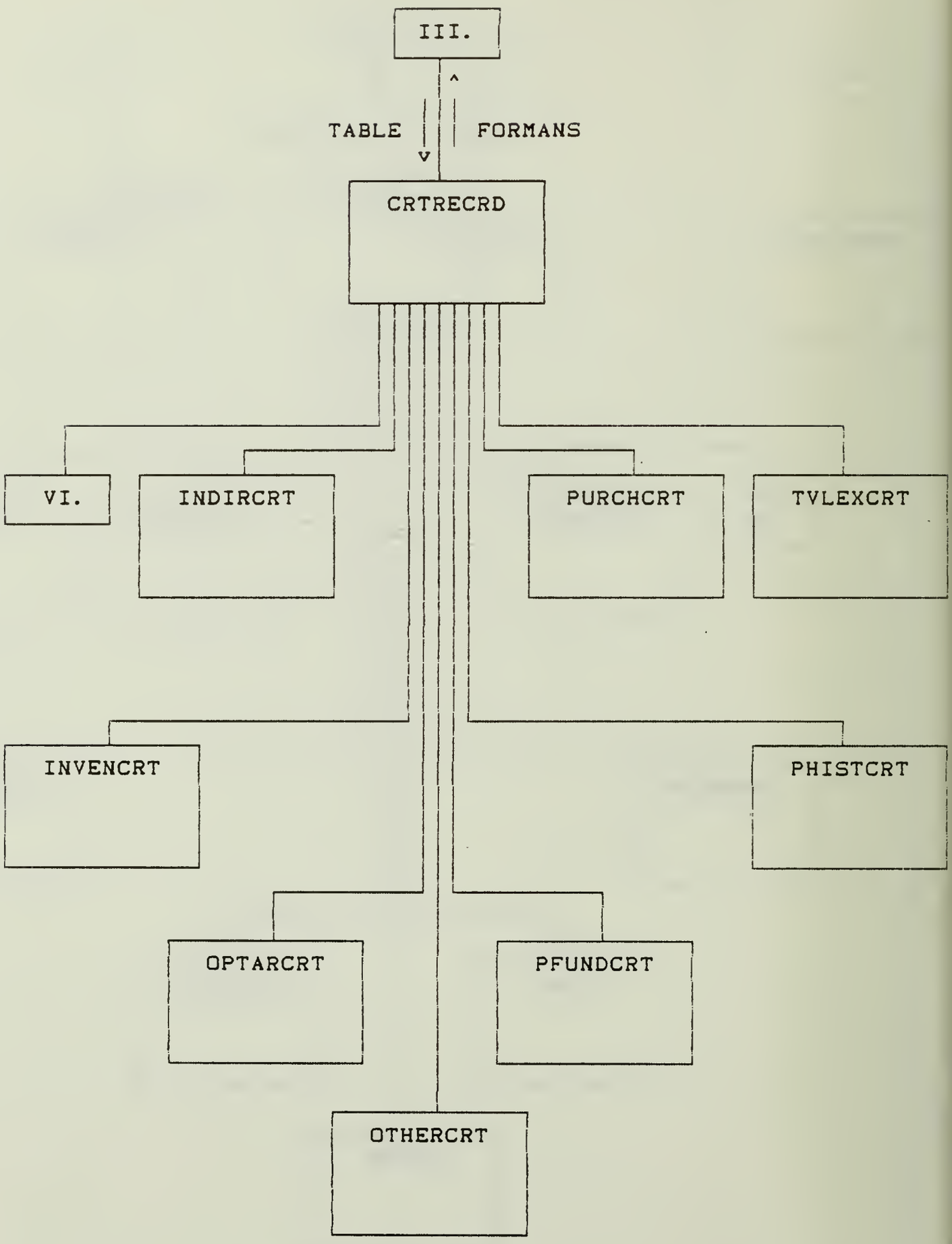
DATA: a. FORMANS

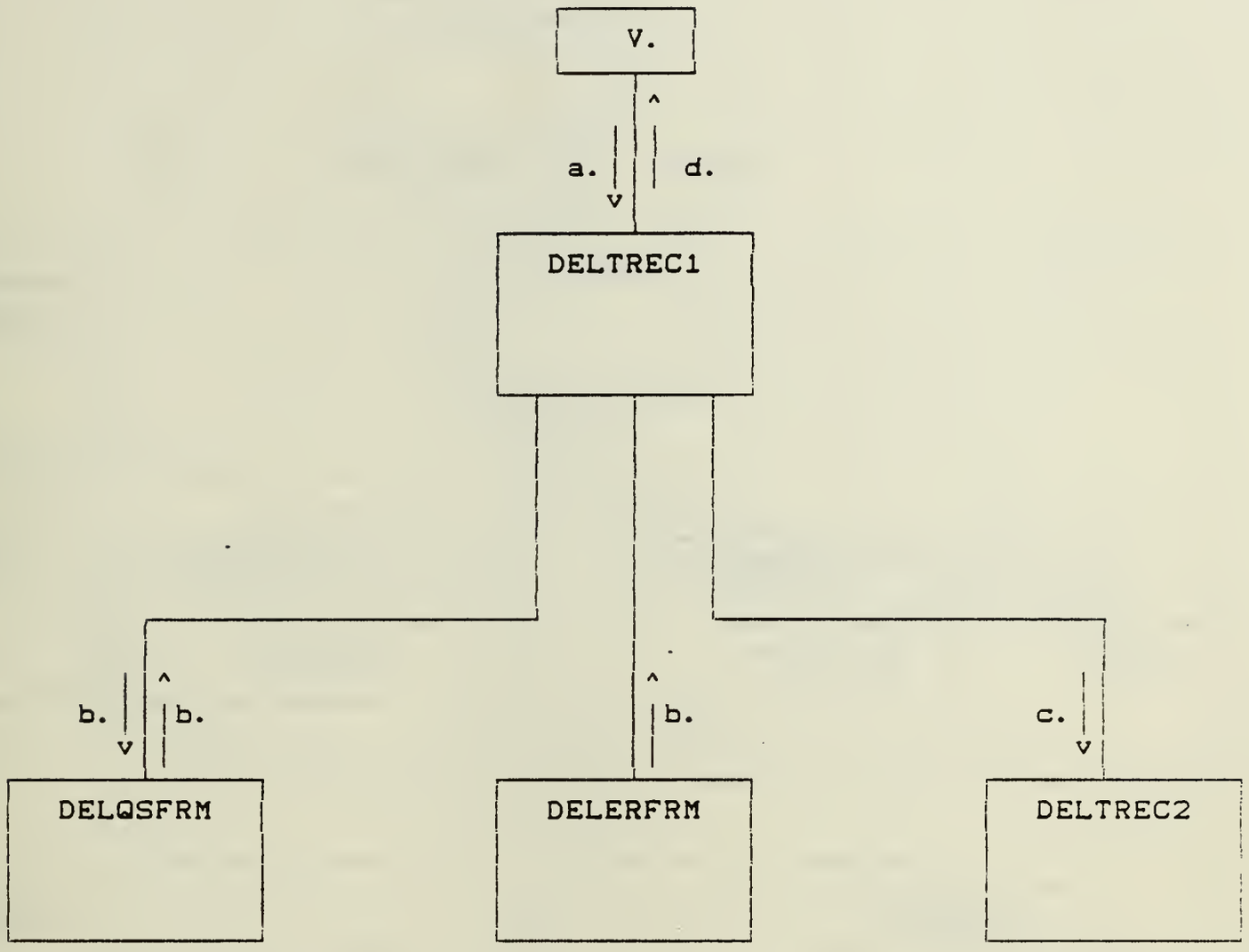


DATA: a. FORMANS

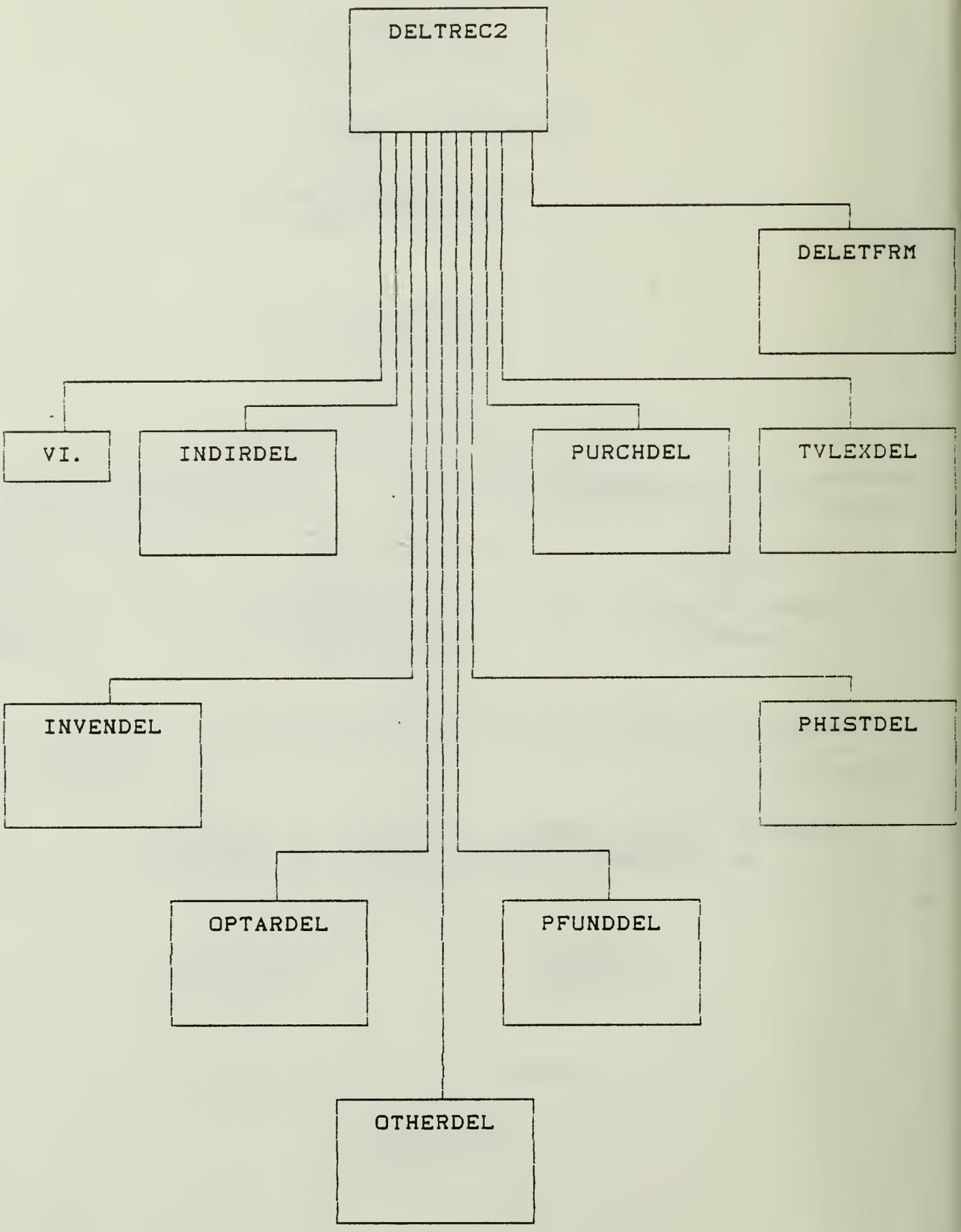

I.
^
FORMANS
ERRORFRM
VI.
TABLDFRM
VII.
TRPLDFRM

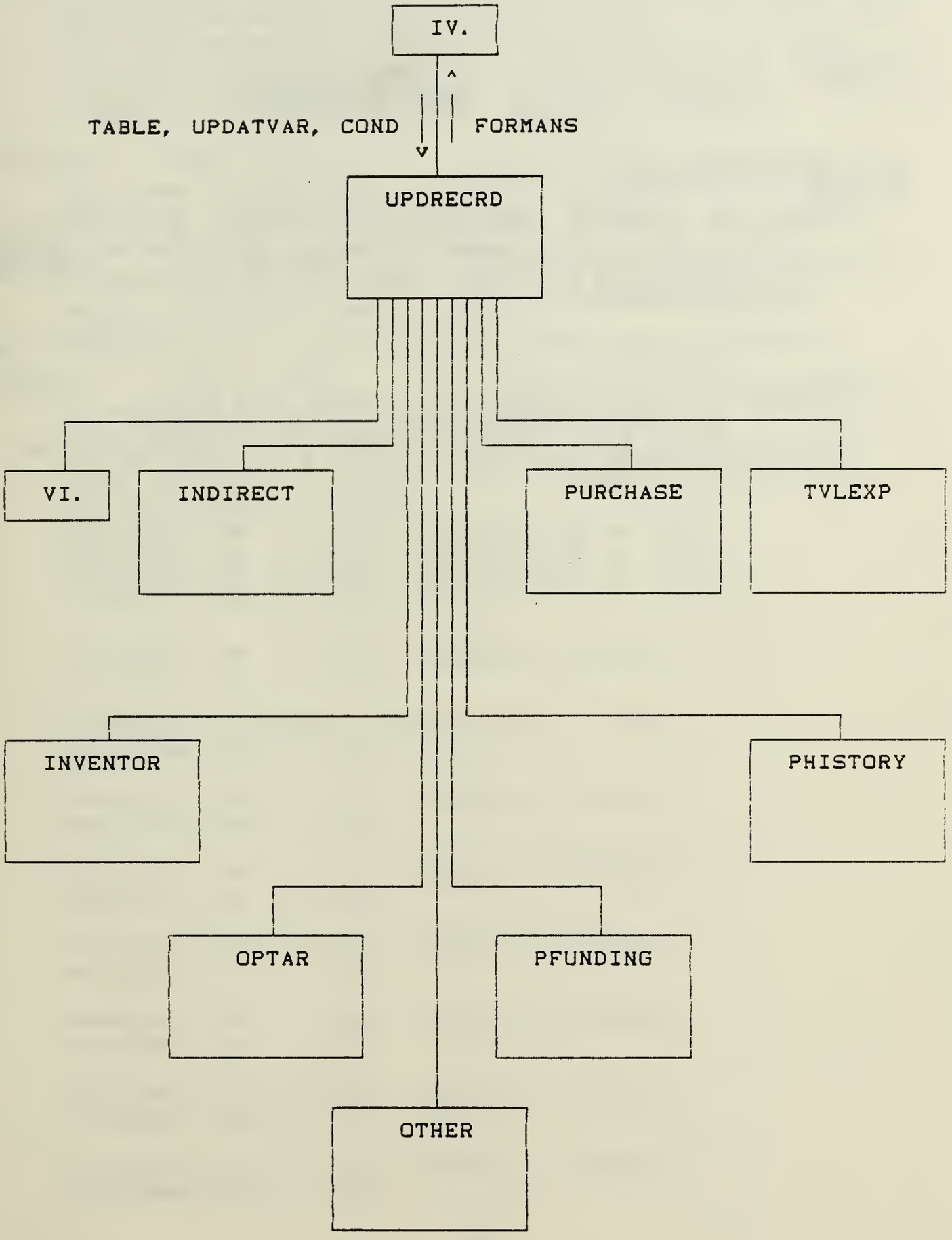

II.
TABLE
FORMANS
BRWTABLE
VI.
INDIRECT
PURCHASE
TVLEXP
INVENTOR
PHISTORY
OPTAR
PFUNDING
OTHER


III.
TABLE
FORMANS
CRTRECRD
VI.
INDIRCRT
PURCHCRT
TVLEXCRT
INVENCRT
PHISTCRT
OPTARCRT
PFUNDCRT
OTHERCRT



DATA: a. TABLE, PASSVAR, PASSCOND; b. DELTANS;
c. PASTABLE, DELTVAR, COND; d. FORMANS


DELTREC2
DELETFRM
VI.
INDIRDEL
PURCHDEL
TVLEXDEL
INVENDEL
PHISTDEL
OPTARDEL
PFUNDDEL
OTHERDEL


IV.
TABLE, UPDATVAR, COND
FORMANS
UPDRECRD
VI.
INDIRECT
PURCHASE
TVLEXP
INVENTOR
PHISTORY
OPTAR
PFUNDING
OTHER

APPENDIX H
DATA DICTIONARY

DATA DICTIONARY

Format of elements: Name, allowable values, format description, files inwhich the element is used (.IPF files unless noted), and read and write access (DBA has table matching access codes to individuals).

Allowable values; num = numeric, str = string, logic = logical.

Format; d = place holder for a digit or a sign.
r = placeholder for any ASCII character.
%5r = rrrrr; repeat the code the number of times as the number

COND NUM 0, 1 or 2 A variable used to determine which condition has been selected by a user for either updating or deleting a record from a table. (0 = only 1 condition displayed by updating or deletion form, 1 = 1st condition was selected and 2 = 2nd condition was selected)
DELTREC1, DELTREC2, INDIRUPD, INVENUPD, OPTARUPD, OTHERUPD, PFUNDUPD, PHISTUPD, PURCHUPD, TVLEXUPD, UPDRECRD

DELTANS STR (Y,N,y,n) A variable which determines if a user is sure that a record is required to be deleted.
DELERFRM, DELERFRM.ICF, DELQSFRM, DELQSFRM.ICF, DELTREC1

DELTVAR STR or NUM A variable which is used to determine which selection criterion is used to delete a record. (DELTVAR equals a specific field's value.)
DELTREC2

ENTRYONE STR or NUM A variable which is updated by entering data onto to a update or deletion selection form or an error form. The only records that will be displayed contains a field's value that matches ENTRYONE. (i.e. ENTRYONE = "Smith" will only display records that contains LNAME = "Smith")
ER1INDIR(.IPF & .ICF), ER2INDIR(.IPF & .ICF),
EROINVEN(.IPF & .ICF), EROPFUND(.IPF & .ICF),
EROOPTAR(.IPF & .ICF), EROPHIST(.IPF & .ICF),
EROTVLEX(.IPF & .ICF),

INDELFRM(.IPF & .ICF), INDIRDLT, INDIRUPD,
INUPDFRM(.IPF & .ICF),

INVENDLT, INVENUPD, IVDELFRM(.IPF & .ICF),
IVUPDFRM(.IPF & .ICF),

OPDELFRM(.IPF & .ICF), OPTARDLT, OPTARUPD,
OPUPDFRM(.IPF & .ICF),

OTDELFRM(.IPF & .ICF), OTHEHDLT, OTHERUPD,
OTUPDFRM(.IPF & .ICF),

PFDELFRM(.IPF & .ICF), PFUNDDLT, PFUNDUPD,
PFUPDFRM(.IPF & .ICF),

PHDELFRM(.IPF & .ICF), PHISTDLT, PHISTUPD,
PHUPDFRM(.IPF & .ICF),

PUDELFRM(.IPF & .ICF), PURCHDLT, PURCHUPD,
PUUPDFRM(.IPF & .ICF),

TVDELFRM(.IPF & .ICF), TVLEXDLT, TVLEXUPD,
TVUPDFRM(.IPF & .ICF)

ENTRYTWO STR or NUM A variable which is updated by entering data onto to a update or deletion selection form or an error form. The only records that will be displayed contains a field's value that matches ENTRYONE (i.e. ENTRYONE = "Smith" will only display records that contains LNAME = "Smith"). Used if there are two fields which may be used for updates or deletions.)
ER2INDIR(.IPF & .ICF), EROINVEN(.IPF & .ICF),
EROPFUND(.IPF & .ICF), EROOPTAR(.IPF & .ICF),

INDIRUPD, INUPDFRM(.IPF & .ICF),

INVENDLT, INVENUPD, IVDELFRM(.IPF & .ICF),
IVUPDFRM(.IPF & .ICF),

OPDELFRM(.IPF & .ICF), OPTARDLT, OPTARUPD,
OPUPDFRM(.IPF & .ICF),

OTHERUPD, OTUPDFRM(.IPF & .ICF),
PFDELFRM(.IPF & .ICF), PFUNDDLT, PFUNDUPD,
PFUPDFRM(.IPF & .ICF)

FORMANS STR (1, 2, 3, 4 or 5) The displayed menu's option that is selected by the user.
BRWTABLE, CRTRECRD, DBFORM(.IPF & .ICF), DBMEN, DELTREC1,
ERRORFRM(.IPF & .ICF),
EXPENFRM(.IPF & .ICF), EXPENMEN,
INCOMFRM(.IPF & .ICF), INCOMMEN,

INDIRDLT, INDIRFRM(.IPF & .ICF), INDIRMEN, INDIRUPD,

INVENDLT, INVENFRM(.IPF & .ICF), INVENMEN, INVENRPT,
INVENUPD, MAINFRM(.IPF & .ICF), MAINMEN,

OPTARDLT, OPTARFRM(.IPF & .ICF), OPTARMEN, OPTARUPD,
OTHERDLT, OTHERFRM(.IPF & .ICF), OTHERMEN, OTHERUPD,
OUTSTRPT, PERSDMEN,

PFUNDDLT, PFUNDFRM(.IPF & .ICF), PFUNDMEN, PFUNDRPT,
PFUNDUPD,

PHISTDLT, PHISTFRM(.IPF & .ICF), PHISTMEN, PHISTRPT,
PHISTUPD, PRPRTMEN,

PURCHDLT, PURCHFRM(.IPF & .ICF), PURCHMEN, PURCHUPD,

PURRTFRM(.IPF & .ICF), PURRTMEN, REPRTFRM(.IPF & .ICF),
REPRTMEN, REQUSRPT,

TRAVFRM(.IPF & .ICF), TRAVLMEN, TVLEXDLT, TVLEXUPD,
UPRECRD

```
INDIRECT.ASAPPRO    NUM  $ddddd.dd
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : A.............
   Write Access : A.............


INDIRECT.AUTH       NUM  $ddddddd.dd  Total amount authorized
     for research project.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.COSTCODE  STR  %5r         Costcode or Job Order #.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF, UPDRECRD
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.DEPTIC     NUM  $ddddd.dd  Amount anticipated for
     indirect costs.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.DOC#       STR  %15r       Document #.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : A.............
   Write Access : A.............


INDIRECT.ENTRY      NUM dd-dd-dd    Date the entry is made.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.ESTIMATE  NUM  $ddddddd.dd  Estimated cost of research.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

```
INDIRECT.EXPIRE    NUM  dd-dd-dd   Date of expipation of funds.
     DELTREC2, INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.ICRECD    NUM  $ddddd.dd  Actual indirect costs re-
     ceived from project.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.MIPR#     STR  %15r       MIPR# from acceptance docu-
     ment.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.PERIOD    STR  %20r       Dates specified on proposal
     for performance of research.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.PRINCIPL  STR  %20r       Last name of principal inves-
     tigator.
     BRWTABLE, INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF, UPDRECRD
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.PROPOSAL  STR  %130r      Title of proposal.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

```
INDIRECT.REF#       STR  %25r       Reference # from acceptance
     document.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.REMARKS    STR  %50r       Remarks.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.SEGMENT    STR  %6r        Segment #
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : A...............
   Write Access : A...............


INDIRECT.SERIALS    NUM  dddd-dddd  Serial  #
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : A...............
   Write Access : A...............


INDIRECT.SPONSOR    STR  %40r       Name of project sponsor.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.STAFFLBR   NUM  $ddddd.dd  Total of support labor.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


INDIRECT.STATUS     STR  %25r       Status of proposal.
     INDIRECT.ITB, INDIRECT, INDIRECT.ICF, INDIRCRT,
     INDIRCRT.ICF, INDIRDEL, INDIRDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

```
INVENTOR.COST       NUM  $dddddd.dd
     INVENTOR.ITB, INVENTOR, INVENTOR.ICF, INVENCRT,
     INVENCRT.ICF, INVENDEL, INVENDEL.ICF, INVENRPT.TPL
   Read  Access : A.............
   Write Access : A.............


INVENTOR.DESCRIPT   STR  %50r
     INVENTOR.ITB, INVENTOR, INVENTOR.ICF, INVENCRT,
     INVENCRT.ICF, INVENDEL, INVENDEL.ICF, INVENRPT.TPL
   Read  Access : A.............
   Write Access : A.............


INVENTOR.ISSUE      STR  %25r
     INVENTOR.ITB, INVENTOR, INVENTOR.ICF, INVENCRT,
     INVENCRT.ICF, INVENDEL, INVENDEL.ICF, INVENRPT.TPL
   Read  Access : A.............
   Write Access : A.............


INVENTOR.LOCATION   STR  %7r
     INVENTOR.ITB, INVENTOR, INVENTOR.ICF, INVENCRT,
     INVENCRT.ICF, INVENDEL, INVENDEL.ICF, INVENRPT.TPL
   Read  Access : A.............
   Write Access : A.............


INVENTOR.PA#        STR  %6r
     BRWTABLE, DELTREC2, INVENTOR.ITB, INVENTOR, INVENTOR.ICF,
     INVENCRT, INVENCRT.ICF, INVENDEL, INVENDEL.ICF,
     INVENRPT.TPL, UPDRECRD
   Read  Access : A.............
   Write Access : A.............


INVENTOR.PO#        STR  %9r
     INVENTOR.ITB, INVENTOR, INVENTOR.ICF, INVENCRT,
     INVENCRT.ICF, INVENDEL, INVENDEL.ICF, INVENRPT.TPL
   Read  Access : A.............
   Write Access : A.............


INVENTOR.RECEIVED   NUM  dd-dd-dd
     INVENTOR.ITB, INVENTOR, INVENTOR.ICF, INVENCRT,
     INVENCRT.ICF, INVENDEL, INVENDEL.ICF, INVENRPT.TPL
   Read  Access : A.............
   Write Access : A.............
```

```
INVENTOR.SERIAL#   STR  %12r
     DELTREC2, INVENTOR.ITB, INVENTOR, INVENTOR.ICF, INVENCRT,
     INVENCRT.ICF, INVENDEL, INVENDEL.ICF, INVENRPT.TPL,
     UPDRECRD
   Read  Access : A...............
   Write Access : A...............


INVENTOR.VENDOR    STR  %40r
     INVENTOR.ITB, INVENTOR, INVENTOR.ICF, INVENCRT,
     INVENCRT.ICF, INVENDEL, INVENDEL.ICF, INVENRPT.TPL
   Read  Access : A...............
   Write Access : A...............


OPTAR.ACTUAL       NUM  $dddddd.dd  Changes in authorized.
     OPTAR.ITB, OPTAR, OPTAR.ICF, OPTARCRT, OPTARCRT.ICF,
     OPTARDEL, OPTARDEL.ICF
   Read  Access : A...............
   Write Access : A...............


OPTAR.AUTH         NUM  $dddddd.dd  Total amount authorized.
     OPTAR.ITB, OPTAR, OPTAR.ICF, OPTARCRT, OPTARCRT.ICF,
     OPTARDEL, OPTARDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


OPTAR.DIFFER    Virtual field : ((ACTUAL - AUTH))
     OPTAR.ITB, OPTAR, OPTAR.ICF, OPTARCRT, OPTARCRT.ICF,
     OPTARDEL, OPTARDEL.ICF
   Read  Access : A...............


OPTAR.ENTRY        NUM  dd-dd-dd    Date that record is entered.
     DELTREC2, OPTAR.ITB, OPTAR, OPTAR.ICF, OPTARCRT,
     OPTARCRT.ICF, OPTARDEL, OPTARDEL.ICF, UPDRECRD
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


OPTAR.RECEIVED     NUM  dd-dd-dd    Date of latest authorization
     memo.
     BRWTABLE, DELTREC2, OPTAR.ITB, OPTAR, OPTAR.ICF, OPTARCRT,
     OPTARCRT.ICF, OPTARDEL, OPTARDEL.ICF, UPDRECRD
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

```
OPTAR.TACTUAL      NUM  $dddddd.dd  Changes in travel authorized.
     OPTAR.ITB, OPTAR, OPTAR.ICF, OPTARCRT, OPTARCRT.ICF,
     OPTARDEL, OPTARDEL.ICF
   Read  Access : A..............
   Write Access : A..............


OPTAR.TAUTH        NUM  $dddddd.dd  Amount designated for travel.
     OPTAR.ITB, OPTAR, OPTAR.ICF, OPTARCRT, OPTARCRT.ICF,
     OPTARDEL, OPTARDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


OPTAR.TDIFFER   Virtual field : ((TACTUAL - TAUTH))
     OPTAR.ITB, OPTAR, OPTAR.ICF, OPTARCRT, OPTARCRT.ICF,
     OPTARDEL, OPTARDEL.ICF
   Read  Access : A..............


OTHER.AUTH         NUM $dddddd.dd   Amount authorized for
     Department Expenditure.
     OTHER.ITB, OTHER, OTHER.ICF, OTHERCRT,
     OTHERCRT.ICF, OTHERDEL, OTHERDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


OTHER.COSTCODE     STR  %5r          Costcode or Job Order #.
     OTHER.ITB, OTHER, OTHER.ICF, OTHERCRT,
     OTHERCRT.ICF, OTHERDEL, OTHERDEL.ICF, UPDRECRD
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


OTHER.ENTRY        NUM   dd-dd-dd   Date the entry is made.
     OTHER.ITB, OTHER, OTHER.ICF, OTHERCRT,
     OTHERCRT.ICF, OTHERDEL, OTHERDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


OTHER.EXPIRE       NUM  dd-dd-dd    Date of expiration of funds.
     DELTREC2, OTHER.ITB, OTHER, OTHER.ICF, OTHERCRT,
     OTHERCRT.ICF, OTHERDEL, OTHERDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

OTHER.PRINCIPL STR %20r Last name of principal invesigator.
BRWTABLE, OTHER.ITB, OTHER, OTHER.ICF, OTHERCRT, OTHERCRT.ICF, OTHERDEL, OTHERDEL.ICF, UPDRECRD
Read Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP

OTHER.PROPOSAL STR %130r Title of proposal.
OTHER.ITB, OTHER, OTHER.ICF, OTHERCRT, OTHERCRT.ICF, OTHERDEL, OTHERDEL.ICF
Read Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP

OTHER.REMARKS STR %50r REMARKS
OTHER.ITB, OTHER, OTHER.ICF, OTHERCRT, OTHERCRT.ICF, OTHERDEL, OTHERDEL.ICF
Read Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP

OTHER.SPONSOR STR %35r Name of project sponsor.
OTHER.ITB, OTHER, OTHER.ICF, OTHERCRT, OTHERCRT.ICF, OTHERDEL, OTHERDEL.ICF
Read Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP

OTHER.STATUS STR %25r Status of funds.
OTHER.ITB, OTHER, OTHER.ICF, OTHERCRT, OTHERCRT.ICF, OTHERDEL, OTHERDEL.ICF
Read Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP

PASSCOND NUM 0, 1 or 2 A variable used to determine which condition has been selected by a user for deleting a record from a table. (0 = only 1 condition displayed by deletion form, 1 = 1st condition was selected and 2 = 2nd condition was selected)
DELTREC1, INDIRDLT, INVENDLT, OPTARDLT, OTHERDLT, PFUNDDLT, PHISTDLT, PURCHDLT, TVLEXDLT

PASSVAR STR or NUM A variable which is used to determine which selection criterion is used to delete a record. (DELTVAR equals a specific field's value.)
DELTREC1, INDIRDLT, INVENDLT, OPTARDLT, OTHERDLT, PFUNDDLT, PHISTDLT, PURCHDLT, TVLEXDLT

```
PASTABLE     STR                    The name of a table that
     have a record deleted from it.
     DELTREC1, DELTREC2


PFUNDING.AMOUNT    NUM  $ddddd.dd  $Amount charged to costcode.
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF, PFUNDRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PFUNDING.CODE      STR  %3r        Department Code
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF
   Read  Access : A...............
   Write Access : A...............


PFUNDING.COSTCODE  STR  %5r        Costcode or Job Order#
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF, PFUNDRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PFUNDING.DAYS      STR  %5r        # of days.
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF, PFUNDRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PFUNDING.ENTRY     NUM  dd-dd-dd   Date of item entry.
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PFUNDING.FNAME     STR  %15r       Full first name.
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF, PFUNDRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

```
PFUNDING.FUNDFROM  NUM  dd-dd-dd   Date funding began.
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF, PFUNDRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PFUNDING.FUNDTO    NUM  dd-dd-dd   Date funding ends.
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF, PFUNDRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PFUNDING.HOURS     NUM d
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF, PFUNDRPT.TPL
   Read  Access : A...............
   Write Access : A...............


PFUNDING.HRSWK     NUM dd          Hours per week.
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF, PFUNDRPT.TPL
   Read  Access : A...............
   Write Access : A...............


PFUNDING.LABOR#    NUM  dd-dd-dd   Date of funding memo.
     DELTREC2, PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF, UPDRECRD
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PFUNDING.LNAME     STR  %25r       Full last name.
     BRWTABLE, DELTREC2, PFUNDING.ITB, PFUNDING, PFUNDING.ICF,
     PFUNDCRT, PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF,
     PFUNDRPT.TPL, UPDRECRD
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PFUNDING.RATE      NUM $ddd.dd     Pay rate.
     PFUNDING.ITB, PFUNDING, PFUNDING.ICF, PFUNDCRT,
     PFUNDCRT.ICF, PFUNDDEL, PFUNDDEL.ICF, PFUNDRPT.TPL
   Read  Access : A...............
   Write Access : A...............
```

```
PHISTORY.BILLET#   STR  %4r        Billet# assigned by CPO.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF, PHISTRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.DOB       NUM  dd-dd-dd   Date of birth.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.EFFECTIV  NUM  dd-dd-dd   Effective date of personnel
     changes.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.ENTRY     NUM  dd-dd-dd   Date of entry.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.ES        NUM   dd-dd-dd  Date elements and standards
     set for performance review.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.EXPIRE    NUM   dd-dd-dd  Date of expiration of tem-
     porary appointments.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF, PHISTRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.FNAME     STR  %15r       Full first name.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF, PHISTRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

```
PHISTORY.GRADE      NUM  dd          Grade level.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF, PHISTRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.HRSWK      NUM   dd          # of hours worked per week.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.JOBTITLE   STR  %30r       Title of position.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF, PHISTRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.LNAME      STR  %25r       Full last name.
     BRWTABLE, DETREC2, PHISTORY.ITB, PHISTORY, PHISTORY.ICF,
     PHISTCRT, PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF,
     PHISTRPT.TPL, UPDRECRD
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.PD#        STR  %5r        Position description #
      assigned by CPO.
      PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
      PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.RATE             $ddd.dd    Hourly rate.
Virtual field : (((SALARY / 2087) * 8) * 1.30000)
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP


PHISTORY.REMARKS    STR  %100r      Remarks.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

```
PHISTORY.SALARY     NUM  $ddddd.dd  Annual salary.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.SERIES     NUM  ddddd      Series classification.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF, PHISTRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.STATUS     STR  %6r        Type of appointment.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.STEP       NUM  dd         Step level.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF, PHISTRPT.TOL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.SUPERVSR   STR  %25r       Last name of workload super-
     visor.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF, PHISTRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PHISTORY.WORKUNIT   STR   %10r      Workunit.
     PHISTORY.ITB, PHISTORY, PHISTORY.ICF, PHISTCRT,
     PHISTCRT.ICF, PHISTDEL, PHISTDEL.ICF, PHISTRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.CODE         STR  %3r      Department code. (See data
     sources in User's Manual.)
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

```
PURCHASE.COSTCODE     STR  %5r       Costcode or JON.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF, REQUSRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP

PURCHASE.DESCRIPT     STR  %50r      Item description.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF, REQUSRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.DOC#         STR  %10r
     BRWTABLE, DELTREC2, PURCHASE.ITB, PURCHASE, PURCHASE.ICF,
     PURCHCRT, PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF,
     REQUSRPT.TPL, UPDRECRD
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.ENTRY        NUM dd-dd-dd  Date record is entered.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.ESTIMATE     NUM $ddddd.dd Estimate of cost.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.FIRM         NUM $ddddd.dd Actual cost of item.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.ISSUE        STR  %25r     Last name of individual item
     is issued to.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

```
PURCHASE.LOCATION    STR  %7r      Location of issue.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.PA#         STR  %17r     Plant Account #.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.PO#         STR  %9r      Purchase Order #.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF, REQUSRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.PRIORITY    NUM  dd
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.QTY         STR  rrr
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : A..............
   Write Access : A..............


PURCHASE.RDD         NUM dd-dd-dd  Required delivery date.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF, REQUSRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.RECEIVED    NUM  dd-dd-dd
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

```
PURCHASE.REMARKS      STR  %50r
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.REQUESTR     STR  %25r      Last name of individual who
     requested the purchase.
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.SERIAL#      STR  %12r
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.STOCK#       STR  %20r
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF, REQUSRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


PURCHASE.VENDOR       STR  %40r
     PURCHASE.ITB, PURCHASE, PURCHASE.ICF, PURCHCRT,
     PURCHCRT.ICF, PURCHDEL, PURCHDEL.ICF, REQUSRPT.TPL
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


TABLE         STR                    The name of a table that
     one of the following functions will be performed on;
     browse the table, update a record in the table, create
     or delete a record.
     BRWTABLE, CRTRECRD, DELTREC1
     INDIRDLT, INDIRMEN, INDIRUPD,
     INVENDLT, INVENMEN, INVENUPD,
     OPTARDLT, OPTARMEN, OPTARUPD,
     OTHERDLT, OTHERMEN, OTHERUPD,
     PFUNDDLT, PFUNDMEN, PFUNDUPD,
     PHISTDLT, PHISTMEN, PHISTUPD,
     PURCHDLT, PURCHMEN, PURCHUPD,
     TRAVLMEN, TVLEXDLT, TVLEXUPD,
     UPDRECRD
```

```
TVLEXP.ADVANCE      NUM  $ddd.dd    $Amount of advance.
     TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
     TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


TVLEXP.CLAIM        NUM  dd-dd-dd   Date claim is filed.
     OUTSTRPT.TPL, TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
     TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


TVLEXP.CODE         STR  %3r        Code indicating type of
     travel; recruit, invitational, regular, or association.
     TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
     TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


TVLEXP.COSTCODE     STR  %5r        Costcode or job order#.
     OUTSTRPT.TPL, TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
     TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


TVLEXP.DATEDEP      NUM  dd-dd-dd   Date travel originated.
     BRWTABLE, TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
     TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


TVLEXP.DATERET      NUM  dd-dd-dd   Date travel is concluded.
     TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
     TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP


TVLEXP.DOC#         STR  %10r       Travel order #.
     BRWTABLE, DELTREC2, OUTSTRPT.TPL, TVLEXP.ITB, TVLEXP,
     TVLEXP.ICF, TVLEXCRT, TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF,
     UPDRECRD
   Read  Access : ABCDEFGHIJKLMNOP
   Write Access : ABCDEFGHIJKLMNOP
```

TVLEXP.DOV# STR rrrrr Disbursing Office Voucher #.
OUTSTRPT.TPL, TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
Read Access : A.....
Write Access : A.....

TVLEXP.ENTRY NUM dd-dd-dd Date entry of record.
TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
Read Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP

TVLEXP.ESTIMATE NUM $dddd.dd Estimated cost of travel.
TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
Read Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP

TVLEXP.FIRM NUM $ddddd.dd Actual cost of travel.
TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
Read Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP

TVLEXP.LOCATION STR %25r Destination of traveler.
TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
Read Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP

TVLEXP.REQUESTR STR %20r Name of travler.
OUTSTRPT.TPL, TVLEXP.ITB, TVLEXP, TVLEXP.ICF, TVLEXCRT,
TVLEXCRT.ICF, TVLEXDEL, TVLEXDEL.ICF
Read Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP

UPDATVAR STR or NUM A variable which is used to
determine which selection criterion is used to update a
record. (UPDATVAR equals a specific field's value.)
INDIRUPD, INVENUPD, OPTARUPD, OTHERUPD, PFUNDUPD, PHISTUPD,
PURCHUPD, TVLEXUPD, UPDRECRD

APPENDIX I
PROGRAM LISTINGS

TABLE OF CONTENTS

This appendix contains listings of all the AS System's perform files, except menus and forms. The code for the menus and forms can be "looked at" by using K-Paint.

```
/* BRWTABLE.IPF -  This module browses a table.                    */
/* Called by:  INDIRMEN.IPF, OPTARMEN.IPF, PFUNDMEN.IPF,          */
/*             PHISTMEN.IPF, PURCHMEN.IPF, TRAVLMEN.IPF,          */
/*             OTHERMEN.IPF, INVENMEN.IPF                         */
/* Calls:      INDIRECT.IPF, OPTAR.IPF, PFUNDING.IPF,             */
/*             PHISTORY.IPF, PURCHASE.IPF, TVLEXP.IPF,            */
/*             OTHER.IPF, TABLDFRM.IPF, INVENTOR.IPF              */
/* Author: R. Booker, 15 Oct 85                                   */

E.ICAS = TRUE;  ! ignore case differences among string values
PERFORM TABLDFRM;
PUTFORM TABLDFRM;

TABLE = #A;

TEST TABLE
    CASE "INDIRECT":
         USE INDIRECT;
         SORT INDIRECT BY AZ PRINCIPL;
         LOAD FROM INDIRECT;   ! the indirect funds entry form
         BROWSE INDIRECT ALL WITH INDIRECT;
         BREAK;
    CASE "INVENTOR":
         USE INVENTOR;
         SORT INVENTOR BY AZ PA#;
         LOAD FROM INVENTOR;   ! the inventory entry form
         BROWSE INVENTOR ALL WITH INVENTOR;
         BREAK;
    CASE "OPTAR":
         USE OPTAR;
         SORT OPTAR BY AZ RECEIVED;
         LOAD FROM OPTAR;   ! the OPTAR entry form
         BROWSE OPTAR ALL WITH OPTAR;
         BREAK;
```

```
    CASE "OTHER":
         USE OTHER;
         SORT OTHER BY AZ PRINCIPL;
         LOAD FROM OTHER;    ! the other funds entry form
         BROWSE OTHER ALL WITH OTHER;
         BREAK;
    CASE "PFUNDING":
         USE PFUNDING;
         SORT PFUNDING BY AZ LNAME;
         LOAD FROM PFUNDING; ! the personnel funding entry form
         BROWSE PFUNDING ALL WITH PFUNDING;
         BREAK;
    CASE "PHISTORY":
         USE PHISTORY;
         SORT PHISTORY BY AZ LNAME;
         LOAD FROM PHISTORY; ! the personnel history entry form
         BROWSE PHISTORY ALL WITH PHISTORY;
         BREAK;
    CASE "PURCHASE":
         USE PURCHASE;
         SORT PURCHASE BY AZ DOC#;
         LOAD FROM PURCHASE;    ! the purchase entry form
         BROWSE PURCHASE ALL WITH PURCHASE;
         BREAK;
    CASE "TVLEXP":
         USE TVLEXP;
         SORT TVLEXP BY AZ DATEDEP, DOC#;
         LOAD FROM TVLEXP;    ! the travel expenses entry form
         BROWSE TVLEXP ALL WITH TVLEXP;
         BREAK;
ENDTEST;
FINISH ALL;
FORMANS = " ";
E.ICAS = FALSE;
RETURN;
```

```
/* CRTRECRD.IPF - This module creates a record in a passed    */
/*                table.                                      */
/* Called by: PFUNDMEN.IPF, PHISTMEN.IPF, TRAVLMEN.IPF,      */
/*            INDIRMEN.IPF, OTHERMEN.IPF, PURCHMEN.IPF,      */
/*            OPTARMEN.IPF, INVENMEN.IPF                     */
/* Calls:     INDIRCRT.IPF, OPTARCRT.IPF, PFUNDCRT.IPF,      */
/*            PHISTCRT.IPF, PURCHCRT.IPF, TVLEXCRT.IPF       */
/*            OTHERCRT.IPF, TABLDFRM.IPF, INVENCRT.IPF       */
/* Author:    R. Booker     26 Jan 86                        */

E.LMOD=FALSE; ! only blanks will first appear in the creation
              ! form
PERFORM TABLDFRM;
PUTFORM TABLDFRM;

TABLE = #A;
TEST TABLE
    CASE "INDIRECT":
         USE INDIRECT;
         LOAD FROM INDIRCRT;   ! the indirect cost form
         CREATE RECORD FOR INDIRECT WITH INDIRCRT;
         BREAK;

      CASE "INVENTOR":
         USE INVENTOR;
         LOAD FROM INVENCRT;   ! the inventory form
         CREATE RECORD FOR INVENTOR WITH INVENCRT;
         BREAK;

    CASE "OPTAR":
         USE OPTAR;
         LOAD FROM OPTARCRT;   ! the optar form
         E.ICOM = TRUE; ! immediate compute of virtual field
         CREATE RECORD FOR OPTAR WITH OPTARCRT;
         E.ICOM = FALSE; BREAK;
```

```
    CASE "OTHER":
         USE OTHER;
         LOAD FROM OTHERCRT;    ! the other funds form
         CREATE RECORD FOR OTHER WITH OTHERCRT;
         BREAK;

    CASE "PFUNDING":
         USE PFUNDING;
         LOAD FROM PFUNDCRT;    ! the personnel funding form
         CREATE RECORD FOR PFUNDING WITH PFUNDCRT;
         BREAK;

    CASE "PHISTORY":
         USE PHISTORY;
         LOAD FROM PHISTCRT; ! the personnel history form
         CREATE RECORD FOR PHISTORY WITH PHISTCRT;
         BREAK;

    CASE "PURCHASE":
         USE PURCHASE;
         LOAD FROM PURCHCRT;    ! the purchase form
         CREATE RECORD FOR PURCHASE WITH PURCHCRT;
         BREAK;

    CASE "TVLEXP":
         USE TVLEXP;
         LOAD FROM TVLEXCRT;    ! the travel expenses form
         CREATE RECORD FOR TVLEXP WITH TVLEXCRT;
         BREAK;

ENDTEST;
FINISH ALL;
FORMANS = " ";          ! places a blank in calling module prompt
E.LMOD=TRUE;            ! default
RETURN;
```

```
/* DBMEN.IPF - Brings up the data base menu, and perform      */
/* (1.0)       either EXPENMEN.IPF, INCOMMEN.IPF or           */
/*             INVENMEN.IPF                                   */
/* Called by:  MAINMEN.IPF                                    */
/* Calls:      DBFRM.IPF, ERRORFRM.IPF, EXPENMEN.IPF,         */
/*             INCOMMEN.IPF, INVENMEN.IPF                     */
/* Author:     R. Booker       15 Jan 86                      */

FORMANS = " ";

WHILE FORMANS NE "4" DO       ! don't exit from module

    PERFORM DBFRM;            ! expenses form
    PUTFORM DBFRM;
    GETFORM DBFRM;

    WHILE NOT (FORMANS IN ["1","2","3","4"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "EXPENMEN.IPF"; BREAK;
         CASE "2": PERFORM "INCOMMEN.IPF"; BREAK;
         CASE "3": PERFORM "INVENMEN.IPF"; BREAK;
    ENDTEST;

ENDWHILE;                 ! formans = "4"

FORMANS = " ";            ! places a blank at calling menu prompt

RETURN;
```

```
/* DELTREC1.IPF - This module gives a user another chance     */
/*                before deleting a record.                   */
/* Called by: INDIRDLT.IPF, OPTARDLT.IPF, OTHERDLT.IPF,       */
/*            PFUNDDLT.IPF, PHISTDLT.IPF, PURCHDLT.IPF,       */
/*            TVLEXDLT.IPF, INVENDLT.IPF                      */
/* Calls:     DELQSFRM.IPF, DELERFRM.IPF, DELTREC2.IPF        */
/* Author: R. Booker, 30 Nov 85                               */

E.ICAS = TRUE;


TABLE = #A; ! table for deletion, passed from calling module

PASSVAR = #B; ! value of condition variable for deletion

PASSCOND = #C; ! condition for deletion, passed from calling
               ! module

DELTANS = " "; ! updated by DELQSFRM

PERFORM DELQSFRM; ! checks to ensure that the user wants to
PUTFORM DELQSFRM; ! delete
GETFORM DELQSFRM;

WHILE NOT (DELTANS IN ["Y","N"]) DO ! check input
    PERFORM DELERFRM;
    PUTFORM DELERFRM;
    GETFORM DELERFRM;
ENDWHILE;

IF DELTANS = "Y" THEN
    PERFORM "DELTREC2.IPF" USING "TABLE", "PASSVAR", "PASSCOND";
ENDIF;
FORMANS = " "; ! places a blank at calling menu prompt
RETURN;
```

```
/* DELTREC2.IPF - This module deletes a record from a passed  */
/*                table.                                      */
/* Called by: DELTREC1.IPF                                    */
/* Calls:     INDIRDEL.IPF, OTHERDEL.IPF, PHISTDEL.IPF,       */
/*            PURCHDEL.IPF, TVLEXDEL.IPF, DELETFRM.IPF,       */
/*            TABLDFRM.IPF, PFUNDDEL.IPF, OPTARDEL.IPF        */
/*            INVENDEL.IPF                                    */
/* Author: R. Booker, 30 Nov 85                               */

E.ICAS = TRUE;     ! ignore case differences among string values


PASTABLE = #A; ! table for deletion, passed from calling module


DELETVAR = #B;     ! condition variable for deletion


COND     = #C; ! condition for deletion, passed from calling
               ! module

PERFORM TABLDFRM;
PUTFORM TABLDFRM;

TEST PASTABLE ! test the table which will be used for deletion

    CASE "INDIRECT":
         USE INDIRECT;
         LOAD FROM INDIRDEL;   ! the indirect cost deletion form
        BROWSE INDIRECT FOR EXPIRE = DELETVAR ALL WITH INDIRDEL;
         BREAK;
```

```
  CASE "INVENTOR":
       USE INVENTOR;
       LOAD FROM INVENDEL;! the inventory deletion form

       IF COND = 1 THEN        ! browse on PA#
            BROWSE INVENTOR FOR PA# = DELETVAR ALL\
                 WITH INVENDEL;
       ELSE
            IF COND = 2 THEN       ! delete on SERIAL#
                BROWSE INVENTOR FOR SERIAL# = DELETVAR ALL\
                     WITH INVENDEL;
            ENDIF;
       ENDIF;
       BREAK;

CASE "OPTAR":
       USE OPTAR;
       LOAD FROM OPTARDEL;  ! the OPTAR deletion form
       IF COND = 1 THEN     ! browse on RECEIVED
            BROWSE OPTAR FOR RECEIVED = DELETVAR ALL\
                 WITH OPTARDEL;
       ELSE

            IF COND = 2 THEN    ! browse on ENTRY
                 BROWSE OPTAR FOR ENTRY = DELETVAR ALL\
                      WITH OPTARDEL;
            ENDIF;
       ENDIF;
       BREAK;

  CASE "OTHER":
       USE OTHER;
       LOAD FROM OTHERDEL;   ! the other funds deletion form
       BROWSE OTHER FOR EXPIRE = DELETVAR ALL WITH OTHERDEL;
       BREAK;
```

```
CASE "PFUNDING":
     USE PFUNDING;
    LOAD FROM PFUNDDEL;! the personnel funding deletion form

     IF COND = 1 THEN        ! browse on LNAME
          BROWSE PFUNDING FOR LNAME = DELETVAR ALL\
               WITH PFUNDDEL;
     ELSE
          IF COND = 2 THEN       ! delete on LABOR#
              BROWSE PFUNDING FOR LABOR# = DELETVAR ALL\
                    WITH PFUNDDEL;
          ENDIF;
     ENDIF;
     BREAK;

CASE "PHISTORY":
     USE PHISTORY;
     LOAD FROM PHISTDEL; ! the personnel history deletion
                         ! form
     BROWSE PHISTORY FOR LNAME = DELETVAR ALL WITH PHISTDEL;
     BREAK;

CASE "PURCHASE":
     USE PURCHASE;
     LOAD FROM PURCHDEL;    ! the purchase deletion form
     BROWSE PURCHASE FOR DOC# = DELETVAR ALL WITH PURCHDEL;
     BREAK;

CASE "TVLEXP":
     USE TVLEXP;
     LOAD FROM TVLEXDEL;    ! the travel expenses deletion
                            ! form
     BROWSE TVLEXP FOR DOC# = DELETVAR ALL WITH TVLEXDEL;
     BREAK;
```

```
ENDTEST;

IF #FOUND = TRUE THEN  ! this prevents the current record from
                       ! being  deleted if a record is not found

    PERFORM DELETFRM;  ! notification that a record is being
    PUTFORM DELETFRM;  ! deleted

    TEST PASTABLE

         CASE "INDIRECT":
              MARK RECORDS IN INDIRECT WITH TRUE CURRENT;
              COMPRESS INDIRECT;
              BREAK;

         CASE "INVENTOR":
              MARK RECORDS IN INVENTOR WITH TRUE CURRENT;
              COMPRESS INVENTOR;
              BREAK;

         CASE "OPTAR":
              MARK RECORDS IN OPTAR WITH TRUE CURRENT;
              COMPRESS OPTAR;
              BREAK;

         CASE "OTHER":
              MARK RECORDS IN OTHER WITH TRUE CURRENT;
              COMPRESS OTHER;
              BREAK;

         CASE "PFUNDING":
              MARK RECORDS IN PFUNDING WITH TRUE CURRENT;
              COMPRESS PFUNDING;
              BREAK;
```

```
        CASE "PHISTORY":
             MARK RECORDS IN PHISTORY WITH TRUE CURRENT;
             COMPRESS PHISTORY;
             BREAK;

        CASE "PURCHASE":
             MARK RECORDS IN PURCHASE WITH TRUE CURRENT;
             COMPRESS PURCHASE;
             BREAK;

        CASE "TVLEXP":
             MARK RECORDS IN TVLEXP WITH TRUE CURRENT;
             COMPRESS TVLEXP;
             BREAK;

    ENDTEST;

ENDIF;

FINISH ALL;
RETURN;
```

```
/* EXPENMEN.IPF - Brings up the expenses menu and performs    */
/* (1.1)          either PERSDMEN.IPF, TRAVLMEN.IPF or       */
/*                PURCHMEN                                   */
/* Called by: DBMEN.IPF                                      */
/* Calls:     EXPENFRM.IPF, ERRORFRM.IPF, PERSDMEN.IPF,      */
/*            TRAVLMEN.IPF, PURCHMEN.IPF                     */
/* Author:    R. Booker          15 Jan 86                   */

FORMANS = " ";

WHILE FORMANS NE "4" DO      ! don't exit module

    PERFORM EXPENFRM;        ! expenses form
    PUTFORM EXPENFRM;
    GETFORM EXPENFRM;

    WHILE NOT (FORMANS IN ["1","2","3","4"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "PERSDMEN.IPF"; BREAK;
         CASE "2": PERFORM "TRAVLMEN.IPF"; BREAK;
         CASE "3": PERFORM "PURCHMEN.IPF"; BREAK;
ENDTEST;

ENDWHILE;      ! formans = "4"

FORMANS = " "; ! places a blank at calling module prompt

RETURN;
```

```
/* INCOMMEN.IPF (1.2)                                              */
/* Brings up the data base income menu, and either performs        */
/* OPTARMEN.IPF, INDIRMEN.IPF or OTHERMEN.IPF                      */
/* Called by: DBMEN.IPF                                            */
/* Calls:     INCOMFRM.IPF, ERRORFRM.IPF, OPTARMEN.IPF,            */
/*            INDIRMEN.IPF, OTHERMEN.IPF                           */
/* Author:    R. Booker          15 Jan 86                         */

FORMANS = " ";

WHILE FORMANS NE "4" DO      ! don't exit module

    PERFORM INCOMFRM;        ! data base menu
    PUTFORM INCOMFRM;
    GETFORM INCOMFRM;

    WHILE NOT (FORMANS IN ["1","2","3","4"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "OPTARMEN.IPF"; BREAK;
         CASE "2": PERFORM "INDIRMEN.IPF"; BREAK;
         CASE "3": PERFORM "OTHERMEN.IPF"; BREAK;
    ENDTEST;

ENDWHILE;            ! formans = "4"

FORMANS = " ";       ! places a blank at calling menu prompt
RETURN;
```

```
/* INDIRDLT.IPF (1.2.2.2)                                          */
/* Brings up the menu for the deletion of a record from the        */
/* indirect funds table.                                           */
/* Called by: INDIRMEN.IPF                                         */
/* Calls:     INDELFRM.IPF, ERlINDIR.IPF, DELTREC1.IPF             */
/* Author: R. Booker     22 Jan 86                                 */

FORMANS  = " ";

ENTRYONE = 0;          ! entered from indelfrm

PASCOND = 0; ! condition for deletion, no choice = 0

WHILE ENTRYONE = 0 DO      ! exit from module

    PERFORM INDELFRM;      ! indirect deletion menu
    PUTFORM INDELFRM;
    GETFORM INDELFRM;

    WHILE ENTRYONE = 0 DO  ! check input
         PERFORM ERlINDIR;
         PUTFORM ERlINDIR;
         GETFORM ERlINDIR;
    ENDWHILE;

    PERFORM "DELTREC1.IPF" USING "\"INDIRECT\"", "ENTRYONE",\
         "PASCOND";

ENDWHILE;              ! formans not blank

ENTRYONE = 0;

FORMANS = " ";         ! places a blank at calling module prompt
RETURN;
```

```
/* INDIRMEN.IPF - Brings up the menu for indirect income table*/
/* (1.2.2)        and offers the following options: browse the*/
/*                INDIRECT table, add, update or delete a     */
/*                record to or from it, and to quit.          */
/* Called by: INCOMMEN.IPF                                    */
/* Calls:     INDIRFRM.IPF, ERRORFRM.IPF, BRWTABLE.IPF,       */
/*            INDIRUPD.IPF, CRTRECRD.IPF, INDIRDLT.IPF        */
/* Author:    R. Booker     15 Jan 86                         */

FORMANS  = " ";
WHILE FORMANS NE "5" DO      ! exit from module

    PERFORM INDIRFRM;        ! indirect cost menu
    PUTFORM INDIRFRM;
    GETFORM INDIRFRM;

    ! check input
    WHILE NOT (FORMANS IN ["1","2","3","4","5"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "BRWTABLE.IPF" USING "\"INDIRECT\"";
                   BREAK;
         CASE "2": PERFORM "INDIRUPD.IPF"; BREAK;
         CASE "3": PERFORM "CRTRECRD.IPF" USING "\"INDIRECT\"";
                   BREAK;
         CASE "4": PERFORM "INDIRDLT.IPF"; BREAK;
    ENDTEST;

ENDWHILE;             ! formans = "5"
FORMANS = " ";        ! places a blank at calling module prompt
RETURN;
```

```
/* INDIRUPD.IPF - Brings up the menu for updating of the      */
/* (1.2.2.1)      indirect funds table.                       */
/* Called by: INDIRMEN.IPF                                    */
/* Calls:     INUPDFRM.IPF, ER2INDIR.IPF, UPDRECRD.IPF        */
/* Author: R. Booker     22 Jan 86                            */

FORMANS  = " ";
ENTRYONE = " "; ! the 1st condition for an update, entered from
                ! inupdfrm
ENTRYTWO = " "; ! the 2nd condition for an update, entered from
                ! inupdfrm
PASCOND = 0;    ! the field to search for the update
WHILE ENTRYONE = " " AND ENTRYTWO = " " DO ! exit from module
    PERFORM INUPDFRM;          ! indirect funding update menu
    PUTFORM INUPDFRM;
    GETFORM INUPDFRM;
    WHILE ENTRYONE = " " AND ENTRYTWO = " " DO  ! check input
         PERFORM ER2INDIR;
         PUTFORM ER2INDIR;
         GETFORM ER2INDIR;
    ENDWHILE;
    IF ENTRYONE NE " " THEN
         PASCOND = 1;                  ! update by cost code
         PERFORM "UPDRECRD.IPF" USING "\"INDIRECT\"",\
              "ENTRYONE", "PASCOND";
    ELSE
         IF ENTRYTWO NE " " THEN
              PASCOND = 2;             ! update by principl
              PERFORM "UPDRECRD.IPF" USING "\"INDIRECT\"",\
                   "ENTRYTWO", "PASCOND";
         ENDIF;
    ENDIF;
ENDWHILE;          ! formans not blank
ENTRYONE = " ";  ENTRYTWO = " ";
FORMANS = " ";   RETURN;
```

```
/* INVENDLT.IPF (1.3.2)                                              */
/* Brings up the menu for the deletion of a record from the          */
/* inventory table.                                                  */
/* Called by: INVENMEN.IPF                                           */
/* Calls:     IVDELFRM.IPF, EROINVEN.IPF, DELTREC1.IPF               */
/* Author: R. Booker     25 Feb 86                                   */

FORMANS  = " ";
ENTRYONE = " ";          ! entered from ivdelfrm
ENTRYTWO = " ";          ! entered from ivdelfrm
PASCOND = 0;             ! condition for deletion
WHILE ENTRYONE = " " AND ENTRYTWO = " " DO ! exit from module
    PERFORM IVDELFRM;        ! inventory deletion menu
    PUTFORM IVDELFRM;
    GETFORM IVDELFRM;
    WHILE ENTRYONE = " " AND ENTRYTWO = " " DO  ! check input
         PERFORM EROINVEN;
         PUTFORM EROINVEN;
         GETFORM EROINVEN;
    ENDWHILE;
    IF ENTRYONE NE " " THEN
         PASCOND = 1;                    ! delete by PA#
         PERFORM "DELTREC1.IPF" USING "\"INVENTOR\"",\
              "ENTRYONE", "PASCOND";
    ELSE
         IF ENTRYTWO NE " " THEN
              PASCOND = 2;           ! delete by SERIAL#
              PERFORM "DELTREC1.IPF" USING "\"INVENTOR\"",\
                   "ENTRYTWO", "PASCOND";
         ENDIF;
    ENDIF;
ENDWHILE;           ! formans not blank
ENTRYONE = " "; ENTRYTWO = " ";
FORMANS = " ";      ! places a blank at calling module prompt
RETURN;
```

```
/* INVENMEN.IPF - Brings up the menu for the inventory table  */
/* (1.3)          and offers the following options: browse the*/
/*                table, update, add, or delete a record, or  */
/*                quit.                                       */
/* Called by: DBMEN.IPF                                       */
/* Calls:     INVENFRM.IPF, ERRORFRM.IPF, BRWTABLE.IPF,       */
/*            INVENUPD.IPF, CRTRECRD.IPF, INVENDLT.IPF        */
/* Author:    R. Booker     25 Feb 86                         */

FORMANS  = " ";

WHILE FORMANS NE "5" DO      ! exit from module

    PERFORM INVENFRM;        ! inventory menu
    PUTFORM INVENFRM;
    GETFORM INVENFRM;

    ! check input
    WHILE NOT (FORMANS IN ["1","2","3","4","5"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "BRWTABLE.IPF" USING "\"INVENTOR\"";
                   BREAK;
         CASE "2": PERFORM "INVENUPD.IPF"; BREAK;
         CASE "3": PERFORM "CRTRECRD.IPF" USING "\"INVENTOR\"";
                   BREAK;
         CASE "4": PERFORM "INVENDLT.IPF"; BREAK;
    ENDTEST;
ENDWHILE;            ! formans = "5"
FORMANS = " ";       ! places a blank at calling module prompt
RETURN;
```

```
/* INVENRPT.IPF (2.2.2)                                          */
/* This module prints the quarterly inventory report.            */
/* Called by: PURRTMEN.IPF                                       */
/* Calls:     TRPLDFRM.IPF                                       */
/* Author:    R. Booker      25 Feb 86                           */

PERFORM TRPLDFRM;  ! Table and Report Loading form
PUTFORM TRPLDFRM;
E.LSTR = 50;           ! Change string length to 50 characters.
E.OPRN = TRUE;         ! Send output to printer.
E.PDEP = 62;           ! Change page depth to 62 lines.
E.PWID = 130;          ! Change page width to 130 character.

USE INVENTOR           ! inventory table.

REPORT "INVENRPT" ORDER BY AZ PA#
                  ! Report was created with K-Report.

FINISH INVENTOR

E.LSTR = 15;        ! Change all environment variables to their
                    ! defaults.
E.OPRN = FALSE;
E.PDEP = 60;
E.PWID = 120;

FORMANS = " ";      ! places a blank at calling module's prompt
RETURN;
```

```
/* INVENUPD.IPF - Brings up the menu for updating of the      */
/* (1.3.1)        inventory table.                            */
/* Called by: INVENMEN.IPF                                    */
/* Calls:     IVUPDFRM.IPF, EROINVEN.IPF, UPDRECRD.IPF        */
/* Author: R. Booker     25 Feb 86                            */

FORMANS  = " ";
ENTRYONE = " "; ! the 1st condition for an update, entered
                ! from ivupdfrm
ENTRYTWO = " "; ! the 2nd condition for an update, entered
                ! from ivupdfrm
PASCOND = 0;    ! the field to search for the update
WHILE ENTRYONE = " " AND ENTRYTWO = " " DO ! exit from module
    PERFORM IVUPDFRM;          ! inventory update menu
    PUTFORM IVUPDFRM;
    GETFORM IVUPDFRM;
    WHILE ENTRYONE = " " AND ENTRYTWO = " " DO  ! check input
         PERFORM EROINVEN;
         PUTFORM EROINVEN;
         GETFORM EROINVEN;
    ENDWHILE;
  IF ENTRYONE NE " " THEN
         PASCOND = 1;                  ! update by PA#
         PERFORM "UPDRECRD.IPF" USING "\"INVENTOR\"",\
              "ENTRYONE", "PASCOND";
    ELSE
         IF ENTRYTWO NE " " THEN
              PASCOND = 2;    ! update by SERIAL#
              PERFORM "UPDRECRD.IPF" USING "\"INVENTOR\"",\
                   "ENTRYTWO", "PASCOND";
         ENDIF;
    ENDIF;
ENDWHILE;         ! formans not blank
ENTRYONE = " ";  ENTRYTWO = " ";
FORMANS = " ";   RETURN;
```

```
/* MAINMEN.IPF (0)                                                    */
/* Brings up the main AS Department data base menu, performs         */
/* DBMEN.IPF or REPRTMEN.IPF                                          */
/* Calls:  MAINFRM.IPF, ERRORFRM.IPF, DBMEN.IPF, REPRTMEN.IPF         */
/* Author: R. Booker          15 Jan 86                               */

FORMANS = " ";

WHILE FORMANS NE "3" DO        ! exit from module

    PERFORM MAINFRM;           ! system banner menu
    PUTFORM MAINFRM;
    GETFORM MAINFRM;

    WHILE NOT (FORMANS IN ["1","2","3"]) DO      ! input check
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "DBMEN.IPF";    BREAK;
         CASE "2": PERFORM "REPRTMEN.IPF"; BREAK;
    ENDTEST;

ENDWHILE;                      ! formans = "3"

RELEASE ALL;

CLEAR;

RETURN;
```

```
/* OPTARDLT.IPF (1.2.1.2)                                          */
/* Brings up the menu for the deletion of a record from the        */
/* OPTAR funding table.                                            */
/* Called by: OPTARMEN.IPF                                         */
/* Calls:     OPDELFRM.IPF, EROOPTAR.IPF, DELTREC1.IPF             */
/* Author: R. Booker     22 Jan 86                                 */

FORMANS  = " ";
ENTRYONE = 0;          ! entered from opdelfrm
ENTRYTWO = 0;          ! entered from opdelfrm
PASCOND = 0;           ! condition for deletion
WHILE ENTRYONE = 0 AND ENTRYTWO = 0 DO      ! exit from module
    PERFORM OPDELFRM;       ! OPTAR deletion menu
    PUTFORM OPDELFRM;
    GETFORM OPDELFRM;
    WHILE ENTRYONE = 0 AND ENTRYTWO = 0 DO  ! check input
         PERFORM EROOPTAR;
         PUTFORM EROOPTAR;
         GETFORM EROOPTAR;
    ENDWHILE;
    IF ENTRYONE NE 0 THEN
         PASCOND = 1;                       ! delete by RECEIVED
         PERFORM "DELTREC1.IPF" USING "\"OPTAR\"", "ENTRYONE",\
              "PASCOND";
    ELSE
         IF ENTRYTWO NE 0 THEN
              PASCOND = 2;            ! delete by ENTRY
              PERFORM "DELTREC1.IPF" USING "\"OPTAR\"",\
                   "ENTRYTWO", "PASCOND";
         ENDIF;
    ENDIF;
ENDWHILE;            ! formans not blank
ENTRYONE = 0; ENTRYTWO = 0;
FORMANS = " ";       ! places a blank at calling module prompt
RETURN;
```

```
/* OPTARMEN.IPF - Brings up the menu for the OPTAR table and  */
/* (1.2.1)        offers the following options: browse the    */
/*                table, update, add, or delete a record, or  */
/*                quit.                                       */
/* Called by: INCOMMEN.IPF                                    */
/* Calls:     OPTARFRM.IPF, ERRORFRM.IPF, BRWTABLE.IPF,       */
/*            OPTARUPD.IPF, CRTRECRD.IPF, OPTARDLT.IPF        */
/* Author:    R. Booker     15 Jan 86                         */

FORMANS  = " ";

WHILE FORMANS NE "5" DO      ! exit from module

    PERFORM OPTARFRM;        ! OPTAR menu
    PUTFORM OPTARFRM;
    GETFORM OPTARFRM;

    ! check input
    WHILE NOT (FORMANS IN ["1","2","3","4","5"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "BRWTABLE.IPF" USING "\"OPTAR\"";
                   BREAK;
         CASE "2": PERFORM "OPTARUPD.IPF"; BREAK;
         CASE "3": PERFORM "CRTRECRD.IPF" USING "\"OPTAR\"";
                   BREAK;
         CASE "4": PERFORM "OPTARDLT.IPF"; BREAK;
    ENDTEST;
ENDWHILE;            ! formans = "5"
FORMANS = " ";       ! places a blank at calling module prompt
RETURN;
```

```
/* OPTARUPD.IPF (1.2.1.1)                                          */
/* Brings up the menu for updating of the OPTAR funding table.*/
/* Called by: OPTARMEN.IPF                                         */
/* Calls:     OPUPDFRM.IPF, EROOPTAR.IPF, UPDRECRD.IPF             */
/* Author: R. Booker     22 Jan 86                                 */

FORMANS  = " ";
ENTRYONE = 0; ! the 1st condition for an update, entered from
              ! opupdfrm
ENTRYTWO = 0; ! the 2nd condition for an update, entered from
              ! opupdfrm
PASCOND = 0; ! the field to search for the update
WHILE ENTRYONE = 0 AND ENTRYTWO = 0 DO      ! exit from module
    PERFORM OPUPDFRM;           ! personnel funding update menu
    PUTFORM OPUPDFRM;
    GETFORM OPUPDFRM;
    WHILE ENTRYONE = 0 AND ENTRYTWO = 0 DO  ! check input
         PERFORM EROOPTAR;
         PUTFORM EROOPTAR;
         GETFORM EROOPTAR;
    ENDWHILE;
    IF ENTRYONE NE 0 THEN
         PASCOND = 1;                   ! update by RECEIVED
         PERFORM "UPDRECRD.IPF" USING "\"OPTAR\"", "ENTRYONE",\
              "PASCOND";
    ELSE
         IF ENTRYTWO NE 0 THEN
              PASCOND = 2;              ! update by ENTRY
              PERFORM "UPDRECRD.IPF" USING "\"OPTAR\"",\
                   "ENTRYTWO", "PASCOND";
         ENDIF;
    ENDIF;
ENDWHILE;            ! formans not blank
ENTRYONE = 0;  ENTRYTWO = 0;
FORMANS = " "; RETURN;
```

```
/* OTHERDLT.IPF (1.2.3.2)                                          */
/* Brings up the menu for the deletion of a record from the        */
/* other funds table.                                              */
/* Called by: OTHERMEN.IPF                                         */
/* Calls:     OTDELFRM.IPF, ER1INDIR.IPF, DELTREC1.IPF             */
/* Author: R. Booker     22 Jan 86                                 */

FORMANS  = " ";

ENTRYONE = 0;          ! entered from otdelfrm

PASCOND = 0; ! condition for deletion, no choice = 0

WHILE ENTRYONE = 0 DO       ! exit from module

    PERFORM OTDELFRM;       ! other deletion menu
    PUTFORM OTDELFRM;
    GETFORM OTDELFRM;

    WHILE ENTRYONE = 0 DO  ! check input
         PERFORM ER1INDIR;          ! same as for indirect table
         PUTFORM ER1INDIR;
         GETFORM ER1INDIR;
    ENDWHILE;

    PERFORM "DELTREC1.IPF" USING "\"OTHER\"", "ENTRYONE",\
         "PASCOND";

ENDWHILE;         ! formans not blank

ENTRYONE = 0;

FORMANS = " ";    ! places a blank at calling module prompt
RETURN;
```

```
/* OTHERMEN.IPF -  Brings up the menu for other income, and   */
/* (1.2.3)         offers the following options: browse the   */
/*                 OTHER table, add, update, or delete a      */
/*                 record, or quit.                           */
/* Called by: INCOMMEN.IPF                                    */
/* Calls:     OTHERFRM.IPF, ERRORFRM.IPF, BRWTABLE.IPF,       */
/*            OTHERUPD.IPF, CRTRECRD.IPF, OTHERDLT.IPF        */
/* Author:    R. Booker           15 Jan 86                   */

FORMANS  = " ";
WHILE FORMANS NE "5" DO      ! exit from module
    PERFORM OTHERFRM;        ! personnel other funds menu
    PUTFORM OTHERFRM;
    GETFORM OTHERFRM;

    ! check input
    WHILE NOT (FORMANS IN ["1","2","3","4","5"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "BRWTABLE.IPF" USING "\"OTHER\"";
                   BREAK;
         CASE "2": PERFORM "OTHERUPD.IPF"; BREAK;
         CASE "3": PERFORM "CRTRECRD.IPF" USING "\"OTHER\"";
                   BREAK;
         CASE "4": PERFORM "OTHERDLT.IPF"; BREAK;
    ENDTEST;

ENDWHILE;            ! formans = "5"

FORMANS = " ";       ! places a blank at calling module prompt
RETURN;
```

```
/* OTHERUPD.IPF - Brings up the menu for updating of the other*/
/* (1.2.3.1)      funds table.                                 */
/* Called by: OTHERMEN.IPF                                     */
/* Calls:     OTUPDFRM.IPF, ER2INDIR.IPF, UPDRECRD.IPF         */
/* Author: R. Booker     22 Jan 86                             */

FORMANS  = " ";
ENTRYONE = " "; ! the 1st condition for an update, entered
                ! from otupdfrm
ENTRYTWO = " "; ! the 2nd condition for an update, entered
                ! from otupdfrm
PASCOND = 0;    ! the field to search for the update
WHILE ENTRYONE = " " AND ENTRYTWO = " " DO ! exit from module
    PERFORM OTUPDFRM;          ! indirect funding update menu
    PUTFORM OTUPDFRM;
    GETFORM OTUPDFRM;
    WHILE ENTRYONE = " " AND ENTRYTWO = " " DO  ! check input
         PERFORM ER2INDIR;      ! same as for the indirect table
         PUTFORM ER2INDIR;
         GETFORM ER2INDIR;
    ENDWHILE;
    IF ENTRYONE NE " " THEN
         PASCOND = 1;                  ! update by cost code
         PERFORM "UPDRECRD.IPF" USING "\"OTHER\"", "ENTRYONE",\
              "PASCOND";
    ELSE
         IF ENTRYTWO NE " " THEN
              PASCOND = 2;    ! update by principl
              PERFORM "UPDRECRD.IPF" USING "\"OTHER\"",\
                   "ENTRYTWO", "PASCOND";
         ENDIF;
    ENDIF;
ENDWHILE;         ! formans not blank
ENTRYONE = " ";  ENTRYTWO = " ";
FORMANS = " ";   RETURN;
```

```
/* OUTSTRPT.IPF (2.3)                                          */
/* This module prints the outstanding TO(s) report.           */
/* Called by: REPRTMEN.IPF                                    */
/* Calls:     TRPLDFRM.IPF                                    */
/* Author:    R. Booker   15 Jan 85                           */

PERFORM TRPLDFRM; ! Table and Report Loading form
PUTFORM TRPLDFRM;

E.LSTR = 25;            ! Change string length to 25 characters.
E.OPRN = TRUE;          ! Send output to printer.
E.PDEP = 62;            ! Change page depth to 62 lines.
E.PWID = 130;           ! Change page width to 130 character.

USE TVLEXP              ! Travel Expenses table.

REPORT "OUTSTRPT" ORDER BY AZ COSTCODE;

FINISH TVLEXP

E.LSTR = 15; ! Change all environment variables to their
             ! defaults.
E.OPRN = FALSE;
E.PDEP = 60;
E.PWID = 120;

FORMANS = " "; ! places a blank at calling module's prompt

RETURN;
```

```
/* PERSDMEN.IPF (1.1.1) - Brings up the personnel menu          */
/* Called by: EXPENMEN.IPF                                      */
/* Calls:     PERSDFRM.IPF, ERRORFRM.IPF, PFUNDMEN.IPF,         */
/*            PHISTMEN.IPF                                      */
/* Author: R. Booker          15 Jan 86                         */

FORMANS = " ";

WHILE FORMANS NE "3" DO      ! don't exit module
    PERFORM PERSDFRM;        ! personnel menu
    PUTFORM PERSDFRM;
    GETFORM PERSDFRM;

    WHILE NOT (FORMANS IN ["1","2","3"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "PFUNDMEN.IPF"; BREAK;
         CASE "2": PERFORM "PHISTMEN.IPF"; BREAK;
    ENDTEST;

ENDWHILE;              ! formans = "3"

FORMANS = " ";         ! places a blank at calling menu prompt
RETURN;
```

```
/* PFUNDDLT.IPF (1.1.1.1.2)                                          */
/* Brings up the menu for the deletion of a record from the          */
/* personnel funding table.                                          */
/* Called by: PFUNDMEN.IPF                                           */
/* Calls:     PFDELFRM.IPF, EROPFUND.IPF, DELTREC1.IPF               */
/* Author: R. Booker     22 Jan 86                                   */

FORMANS  = " ";
ENTRYONE = " ";          ! entered from pfdelfrm
ENTRYTWO = 0;            ! entered from pfdelfrm
PASCOND = 0;             ! condition for deletion
WHILE ENTRYONE = " " AND ENTRYTWO = 0 DO    ! exit from module
    PERFORM PFDELFRM;         ! personnel funding deletion menu
    PUTFORM PFDELFRM;
    GETFORM PFDELFRM;
    WHILE ENTRYONE = " " AND ENTRYTWO = 0 DO  ! check input
         PERFORM EROPFUND;
         PUTFORM EROPFUND;
         GETFORM EROPFUND;
    ENDWHILE;
    IF ENTRYONE NE " " THEN
         PASCOND = 1;         ! delete by last name
         PERFORM "DELTREC1.IPF" USING "\"PFUNDING\"",\
              "ENTRYONE", "PASCOND";
    ELSE
         IF ENTRYTWO NE 0 THEN
              PASCOND = 2;    ! delete by labor #
              PERFORM "DELTREC1.IPF" USING "\"PFUNDING\"",\
                   "ENTRYTWO", "PASCOND";
         ENDIF;
    ENDIF;
ENDWHILE;            ! formans not blank
ENTRYONE = " "; ENTRYTWO = 0;
FORMANS = " ";       ! places a blank at calling module prompt
RETURN;
```

```
/* PFUNDMEN.IPF -  Brings up the menu for personnel funding.  */
/* (1.1.1.1)       The following options then may be selected:*/
/*                 browse the personnel funding table, update,*/
/*                 add or delete a record.                    */
/* Called by: PERSDMEN.IPF                                    */
/* Calls:     PFUNDFRM.IPF, ERRORFRM.IPF, BRWTABLE.IPF,       */
/*            CRTRECRD.IPF, PFUNDUPD.IPF, PFUNDDLT.IPF        */
/* Author: R. Booker           15 Jan 86                      */

FORMANS  = " ";

WHILE FORMANS NE "5" DO      ! exit from module
    PERFORM PFUNDFRM;                ! personnel funding menu
    PUTFORM PFUNDFRM;
    GETFORM PFUNDFRM;

    ! check input
    WHILE NOT (FORMANS IN ["1","2","3","4","5"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "BRWTABLE.IPF" USING "\"PFUNDING\"";
                   BREAK;
         CASE "2": PERFORM "PFUNDUPD.IPF"; BREAK;
         CASE "3": PERFORM "CRTRECRD.IPF" USING "\"PFUNDING\"";
                   BREAK;
         CASE "4": PERFORM "PFUNDDLT.IPF"; BREAK;
    ENDTEST;

ENDWHILE;          ! formans = "5"

FORMANS = " ";    RETURN;
```

```
/* PFUNDRPT.IPF (2.1.1)                                          */
/* This module prints the personnel funding report.              */
/* Called by: PRPRTMEN.IPF                                       */
/* Calls:     TRPLDFRM.IPF                                       */
/* Author:    R. Booker     15 Jan 86                            */

PERFORM TRPLDFRM; ! Table and Report Loading form
PUTFORM TRPLDFRM;

E.LSTR = 25;            ! Change string length to 25 characters
E.OPRN = TRUE;          ! Send output to printer.
E.PDEP = 62;            ! Change page depth to 62 lines.
E.PWID = 130;           ! Change page width to 130 character.

USE PFUNDING            ! Personnel Funding table.

REPORT "PFUNDRPT" ORDER BY AZ PFUNDING.LNAME,\
                  AZ PFUNDING.FUNDFROM, AZ PFUNDING.FUNDTO
                  ! Report was created with K-Report.

FINISH PFUNDING

E.LSTR = 15;        ! Change all environment variables to their
E.OPRN = FALSE;     ! defaults.
E.PDEP = 60;
E.PWID = 120;

FORMANS = " ";      ! places a blank at calling module's prompt
RETURN;
```

```
/* PFUNDUPD.IPF (1.1.1.1.1) - Brings up the menu for updating */
/* of the personnel funding table.                            */
/* Called by: PFUNDMEN.IPF                                    */
/* Calls:     PFUPDFRM.IPF, EROPFUND.IPF, UPDRECRD.IPF        */
/* Author: R. Booker     22 Jan 86                            */

FORMANS  = " ";
ENTRYONE = " "; ! the 1st condition for an update, entered
                ! from pfupdfrm
ENTRYTWO =  0;  ! the 2nd condition for an update, entered
                ! from pfupdfrm
PASCOND = 0;    ! the field to search for the update
WHILE ENTRYONE = " " AND ENTRYTWO = 0 DO ! exit from module
    PERFORM PFUPDFRM;       ! personnel funding update menu
    PUTFORM PFUPDFRM;
    GETFORM PFUPDFRM;
    WHILE ENTRYONE = " " AND ENTRYTWO = 0 DO  ! check input
         PERFORM EROPFUND;
         PUTFORM EROPFUND;
         GETFORM EROPFUND;
    ENDWHILE;
    IF ENTRYONE NE " " THEN
         PASCOND = 1;       ! update by last name
         PERFORM "UPDRECRD.IPF" USING "\"PFUNDING\"",\
              "ENTRYONE", "PASCOND";
    ELSE
         IF ENTRYTWO NE 0 THEN
              PASCOND = 2;       ! update by labor #
              PERFORM "UPDRECRD.IPF" USING "\"PFUNDING\"",\
                   "ENTRYTWO", "PASCOND";
         ENDIF;
    ENDIF;
ENDWHILE;         ! formans not blank
ENTRYONE = " ";  ENTRYTWO = 0;
FORMANS = " ";   RETURN;
```

```
/* PHISTDLT.IPF (1.1.1.2.2)                                          */
/* Brings up the menu for the deletion of a record from              */
/* personnel history table.                                          */
/* Called by: PHISTMEN.IPF                                           */
/* Calls:     PHDELFRM.IPF, EROPHIST.IPF, DELTREC1.IPF               */
/* Author: R. Booker     19 Jan 86                                   */

FORMANS  = " ";

ENTRYONE = " ";          ! entered from phdelfrm

PASCOND = 0; ! condition for deletion, no choice = 0

WHILE ENTRYONE = " " DO      ! exit from module

    PERFORM PHDELFRM;        ! personnel history deletion menu
    PUTFORM PHDELFRM;
    GETFORM PHDELFRM;

    WHILE ENTRYONE = " " DO  ! check input
         PERFORM EROPHIST;
         PUTFORM EROPHIST;
         GETFORM EROPHIST;
    ENDWHILE;

    PERFORM "DELTREC1.IPF" USING "\"PHISTORY\"", "ENTRYONE",\
         "PASCOND";

ENDWHILE;           ! formans not blank

ENTRYONE = " ";

FORMANS = " ";      ! places a blank at calling module prompt
RETURN;
```

```
/* PHISTMEN.IPF - Brings up the menu for the personnel history*/
/* (1.1.1.2)      table.  The following options are then      */
/*                possible: browse a table, update, add or    */
/*                delete a record, and quit.                  */
/* Called by: PERSDMEN.IPF                                    */
/* Calls:     PHISTFRM.IPF, ERRORFRM.IPF, BRWTABLE.IPF,       */
/*            PHISTUPD.IPF, CRTRECRD.IPF, PHISTDLT.IPF        */
/* Author: R. Booker     15 Jan 86                            */

FORMANS  = " ";

WHILE FORMANS NE "5" DO       ! exit from module
    PERFORM PHISTFRM;         ! personnel history menu
    PUTFORM PHISTFRM;
    GETFORM PHISTFRM;

    ! check input
    WHILE NOT (FORMANS IN ["1","2","3","4","5"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "BRWTABLE.IPF" USING "\"PHISTORY\"";
                   BREAK;
         CASE "2": PERFORM "PHISTUPD.IPF"; BREAK;
         CASE "3": PERFORM "CRTRECRD.IPF" USING "\"PHISTORY\"";
                   BREAK;
         CASE "4": PERFORM "PHISTDLT.IPF"; BREAK;
    ENDTEST;

ENDWHILE;           ! formans = "5"

FORMANS = " ";      RETURN;
```

```
/* PHISTRPT.IPF (2.1.2)                                          */
/* This module prints the personnel history report               */
/* Called by: PRPRTMEN.IPF                                       */
/* Calls:     TRPLDFRM.IPF                                       */
/* Author:    R. Booker       15 Jan 86                          */

PERFORM TRPLDFRM; ! Table and Report Loading form
PUTFORM TRPLDFRM;

E.LSTR = 25;            ! Change string length to 25 characters.
E.OPRN = TRUE;          ! Send output to printer.
E.PDEP = 62;            ! Change page depth to 62 lines.
E.PWID = 136;           ! Change page width to 136 character.
USE PHISTORY            ! Personnel History table.

REPORT "PHISTRPT" ORDER BY AZ LNAME
                        ! Report was created with K-Report.
FINISH PHISTORY

E.LSTR = 15;          ! Change all environment variables to
                      ! their defaults.
E.OPRN = FALSE;
E.PDEP = 60;
E.PWID = 120;

FORMANS = " ";        ! places a blank at calling module's prompt
RETURN;
```

```
/* PHISTUPD.IPF - Brings up the menu for updating of the      */
/* (1.1.1.2.1)    personnel history table.                    */
/* Called by: PHISTMEN.IPF                                    */
/* Calls:     PHUPDFRM.IPF, EROPHIST.IPF, UPDRECRD.IPF        */
/* Author: R. Booker     18 Jan 86                            */

FORMANS  = " ";

ENTRYONE = " ";     ! the condition for an update, entered from
                    ! phupdfrm

PASCOND = 0;        ! the field to search for the update;
                    ! 0 = no option

WHILE ENTRYONE = " " DO       ! exit from module

    PERFORM PHUPDFRM;         ! personnel history update menu
    PUTFORM PHUPDFRM;
    GETFORM PHUPDFRM;

    WHILE ENTRYONE = " " DO   ! check input
         PERFORM EROPHIST;
         PUTFORM EROPHIST;
         GETFORM EROPHIST;
    ENDWHILE;

    PERFORM "UPDRECRD.IPF" USING "\"PHISTORY\"", "ENTRYONE",\
         "PASCOND";

ENDWHILE;           ! formans not blank

ENTRYONE = " ";

FORMANS = " ";    ! places a blank at calling module prompt
RETURN;
```

```
/* PRPRTMEN.IPF (2.1) - Brings up the personnel reports menu */
/* Called by: REPRTMEN.IPF                                   */
/* Calls:     PFUNDRPT.IPF, PHISTRPT.IPF, PRPRTFRM.IPF,      */
/*            ERRORFRM.IPF                                   */
/* Author:    R. Booker          15 Jan 86                   */

FORMANS = " ";

WHILE FORMANS NE "3" DO       ! exit from module

    PERFORM PRPRTFRM;         ! personnel menu
    PUTFORM PRPRTFRM;
    GETFORM PRPRTFRM;

    WHILE NOT (FORMANS IN ["1","2","3"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "PFUNDRPT.IPF"; BREAK;
         CASE "2": PERFORM "PHISTRPT.IPF"; BREAK;
    ENDTEST;

ENDWHILE;        ! formans = "3"

FORMANS = " ";  ! places a blank at calling modules prompt
RETURN;
```

```
/* PURCHDLT.IPF (1.1.3.2)                                          */
/* Brings up the menu for the deletion of a record from            */
/* purchase table.                                                 */
/* Called by: PURCHMEN.IPF                                         */
/* Calls:     PUDELFRM.IPF, EROTVLEX.IPF, DELTREC1.IPF             */
/* Author: R. Booker     22 Jan 86                                 */

FORMANS  = " ";

ENTRYONE = " ";          ! entered from pudelfrm

PASCOND = 0; ! condition for deletion, no choice = 0

WHILE ENTRYONE = " " DO      ! exit from module

    PERFORM PUDELFRM;        ! purchase deletion menu
    PUTFORM PUDELFRM;
    GETFORM PUDELFRM;

    WHILE ENTRYONE = " " DO  ! check input
         PERFORM EROTVLEX;   ! same error form as travel
         PUTFORM EROTVLEX;   ! expenses
         GETFORM EROTVLEX;
    ENDWHILE;

    PERFORM "DELTREC1.IPF" USING "\"PURCHASE\"", "ENTRYONE",\
         "PASCOND";

ENDWHILE;        ! formans not blank

ENTRYONE = " ";

FORMANS = " ";  ! places a blank at calling module prompt
RETURN;
```

```
/* PURCHMEN.IPF -  Brings up the menu for purchase, and offers*/
/* (1.1.3)         the following options; browse the table,   */
/*                 update, add, delete a record from it, or   */
/*                 quit.                                      */
/* Called by: EXPENMEN.IPF                                    */
/* Calls:     PURCHFRM.IPF, ERRORFRM.IPF, BRWTABLE.IPF,       */
/*            PURCHUPD, CRTRECRD.IPF, PURCHDLT.IPF            */
/* Author: R. Booker          15 Jan 86                       */

FORMANS  = " ";

WHILE FORMANS NE "5" DO      ! exit from module
    PERFORM PURCHFRM;                ! purchase menu
    PUTFORM PURCHFRM;
    GETFORM PURCHFRM;

    ! check input
    WHILE NOT (FORMANS IN ["1","2","3","4","5"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "BRWTABLE.IPF" USING "\"PURCHASE\"";
                   BREAK;
         CASE "2": PERFORM "PURCHUPD.IPF"; BREAK;
         CASE "3": PERFORM "CRTRECRD.IPF" USING "\"PURCHASE\"";
                   BREAK;
         CASE "4": PERFORM "PURCHDLT.IPF"; BREAK;
    ENDTEST;

ENDWHILE;     ! formans = "5"
FORMANS = " ";! places a blank at calling module prompt
RETURN;
```

```
/* PURCHUPD.IPF - Brings up the menu for updating of the      */
/* (1.1.3.1)      purchase table.                             */
/* Called by: PURCHMEN.IPF                                    */
/* Calls:     PUUPDFRM.IPF, EROTVLEX.IPF, UPDRECRD.IPF        */
/* Author: R. Booker     23 Jan 86                            */

FORMANS  = " ";

ENTRYONE = " ";     ! the condition for an update, entered
                    ! from puupdfrm

PASCOND = 0;        ! the field to search for the update;
                    ! 0 = no option

WHILE ENTRYONE = " " DO       ! exit from module

    PERFORM PUUPDFRM;         ! purchase update menu
    PUTFORM PUUPDFRM;
    GETFORM PUUPDFRM;

    WHILE ENTRYONE = " " DO   ! check input
         PERFORM EROTVLEX;    ! same error form as travel
         PUTFORM EROTVLEX;    ! expenses
         GETFORM EROTVLEX;
    ENDWHILE;

    PERFORM "UPDRECRD.IPF" USING "\"PURCHASE\"", "ENTRYONE",\
         "PASCOND";

ENDWHILE;         ! formans not blank

ENTRYONE = " ";

FORMANS = " ";  ! places a blank at calling module prompt
RETURN;
```

```
/* PURRTMEN.IPF - Brings up the requisition status/quarterly  */
/* (2.2)          inventory reports menu                      */
/* Called by:     REPRTMEN.IPF                                */
/* Also calls:    PURRTFRM.IPF, ERRORFRM.IPF, REQUSRPT.IPF,   */
/*                INVENRPT.IPF                                */
/* Author:     R. Booker       15 Jan 86                      */

FORMANS = " ";

WHILE FORMANS NE "3" DO      ! exit from module

    PERFORM PURRTFRM; ! requisition status/quarterly inventory
    PUTFORM PURRTFRM; ! form
    GETFORM PURRTFRM;

    WHILE NOT (FORMANS IN ["1","2","3"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "REQUSRPT.IPF"; BREAK;
         CASE "2": PERFORM "INVENRPT.IPF"; BREAK;
    ENDTEST;

ENDWHILE;       ! formans = "3"

FORMANS = " ";  ! places a blank at calling modules prompt
RETURN;
```

```
/* REPRTMEN.IPF - Brings up the reports menu, and performs    */
/* (2)            either PRPRTMEN.IPF, PURRTMEN.IPF,          */
/*                OUTSTRPT.IPF                                */
/* Called by: MAINMEN.IPF                                     */
/* Calls:     REPRTFRM.IPF, ERRORFRM.IPF, PRPRTMEN.IPF,       */
/*            PURRTMEN.IPF, OUTSTRPT.IPF                      */
/* Author: R. Booker                  15 Jan 86               */

FORMANS = " ";

WHILE FORMANS NE "5" DO      ! exit from module

    PERFORM REPRTFRM;        ! reports menu
    PUTFORM REPRTFRM;
    GETFORM REPRTFRM;

    WHILE NOT (FORMANS IN ["1","2","4","5"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM PRPRTMEN; BREAK;
         CASE "2": PERFORM PURRTMEN; BREAK;
/*       CASE "3": PERFORM BALACRPT; BREAK;          */
         CASE "4": PERFORM OUTSTRPT; BREAK;
    ENDTEST;

ENDWHILE;        ! formans = "5"

FORMANS = " ";  ! places a blank at calling module's prompt
RETURN;
```

```
/* REQUSRPT.IPF (2.2.1)                                          */
/* This module prints the requisition status report.             */
/* Called by: PURRTMEN.IPF                                       */
/* Calls:     TRPLDFRM.IPF                                       */
/* Author:    R. Booker          15 Jan 86                       */

PERFORM TRPLDFRM; ! Table and Report Loading form
PUTFORM TRPLDFRM;

E.LSTR = 50;            ! Change string length to 30 characters.
E.OPRN = TRUE;          ! Send output to printer.
E.PDEP = 62;            ! Change page depth to 62 lines.
E.PWID = 130;           ! Change page width to 130 character.

USE PURCHASE            ! Purchase table.

REPORT "REQUSRPT"       ! Report was created with K-Report.
FINISH PURCHASE

E.LSTR = 15;        ! Change all environment variables to their
                    ! defaults.
E.OPRN = FALSE;
E.PDEP = 60;
E.PWID = 120;

FORMANS = " ";  ! places a blank at calling module's prompt
RETURN;
```

```
/* TRAVLMEN.IPF - Brings up the menu for travel expenses,     */
/* (1.1.2)        and offers the following options: browse the*/
/*                travel expense table, update, add or delete */
/*                a record from it.                           */
/* Called by: EXPENMEN.IPF                                    */
/* Calls: TRAVFRM.IPF, ERRORFRM.IPF, BRWTABLE.IPF,            */
/*        TVLEXUPD.IPF, CRTRECRD.IPF, TVLEXDLT.IPF            */
/* Author: R. Booker          15 Jan 86                       */

FORMANS  = " ";

WHILE FORMANS NE "5" DO      ! exit from module

    PERFORM TRAVFRM;         ! travel expenses menu
    PUTFORM TRAVFRM;
    GETFORM TRAVFRM;

    ! check input
    WHILE NOT (FORMANS IN ["1","2","3","4","5"]) DO
         PERFORM ERRORFRM;
         PUTFORM ERRORFRM;
         GETFORM ERRORFRM;
    ENDWHILE;

    TEST FORMANS
         CASE "1": PERFORM "BRWTABLE.IPF" USING "\"TVLEXP\"";
                   BREAK;
         CASE "2": PERFORM "TVLEXUPD.IPF"; BREAK;
         CASE "3": PERFORM "CRTRECRD.IPF" USING "\"TVLEXP\"";
                   BREAK;
         CASE "4": PERFORM "TVLEXDLT.IPF"; BREAK;
    ENDTEST;
ENDWHILE;            ! formans = "5"
FORMANS = " ";       ! places a blank at calling module prompt
RETURN;
```

```
/* TVLEXDLT.IPF - Brings up the menu for the deletion of a    */
/* (1.1.2.2)      record from the travel expenses table.     */
/* Called by: TRAVLMEN.IPF                                   */
/* Calls:     TVDELFRM.IPF, EROTVLEX.IPF, DELTREC1.IPF       */
/* Author: R. Booker     22 Jan 86                           */

FORMANS  = " ";

ENTRYONE = " ";          ! entered from tvdelfrm

PASCOND = 0; ! condition for deletion, no choice = 0

WHILE ENTRYONE = " " DO       ! exit from module

    PERFORM TVDELFRM;         ! travel expenses deletion menu
    PUTFORM TVDELFRM;
    GETFORM TVDELFRM;

    WHILE ENTRYONE = " " DO   ! check input
         PERFORM EROTVLEX;
         PUTFORM EROTVLEX;
         GETFORM EROTVLEX;
    ENDWHILE;

    PERFORM "DELTREC1.IPF" USING "\"TVLEXP\"", "ENTRYONE",\
         "PASCOND";

ENDWHILE;          ! formans not blank

ENTRYONE = " ";

FORMANS = " ";    ! places a blank at calling module prompt
RETURN;
```

```
/* TVLEXUPD.IPF - Brings up the menu for updating of the      */
/* (1.1.2.1)      travel expenses table.                      */
/* Called by: TRAVLMEN.IPF                                    */
/* Calls:     TVUPDFRM.IPF, EROTVLEX.IPF, UPDRECRD.IPF        */
/* Author: R. Booker     22 Jan 86                            */

FORMANS  = " ";

ENTRYONE = " ";     ! the condition for an update, entered
                    ! from tvupdfrm

PASCOND = 0;        ! the field to search for the update;
                    ! 0 = no option

WHILE ENTRYONE = " " DO      ! exit from module

    PERFORM TVUPDFRM;        ! travel expenses update menu
    PUTFORM TVUPDFRM;
    GETFORM TVUPDFRM;

    WHILE ENTRYONE = " " DO  ! check input
         PERFORM EROTVLEX;
         PUTFORM EROTVLEX;
         GETFORM EROTVLEX;
    ENDWHILE;

    PERFORM "UPDRECRD.IPF" USING "\"TVLEXP\"", "ENTRYONE",\
         "PASCOND";

ENDWHILE;           ! formans not blank

ENTRYONE = " ";

FORMANS = " ";    ! places a blank at calling module prompt
RETURN;
```

```
/* UPDRECRD.IPF -  This module updates a record in a table.    */
/* Called by:  INDIRUPD.IPF, OPTARUPD.IPF, PFUNDUPD.IPF,       */
/*             PHISTUPD.IPF, PURCHUPD.IPF, TVLEXUPD.IPF,       */
/*             OTHERUPD.IPF, INVENUPD.IPF                      */
/* Calls:      INDIRECT.IPF, OPTAR.IPF, PFUNDING.IPF,          */
/*             PHISTORY.IPF, PURCHASE.IPF, TVLEXP.IPF,         */
/*             OTHER.IPF, TABLDFRM.IPF, INVENTOR.IPF           */
/* Author: R. Booker          18 Jan 86                        */

E.ICAS = TRUE;     ! ignore case differences among string values
PERFORM TABLDFRM;
PUTFORM TABLDFRM;


TABLE = #A;          ! passed from calling module


UPDATVAR = #B;       ! the condition variable for the search

COND = #C;! the condition field for selecting records for update

TEST TABLE
    CASE "INDIRECT":
         USE INDIRECT;
         LOAD FROM INDIRECT;      ! the indirect funds entry form
         IF COND = 1 THEN         ! browse on COSTCODE
              BROWSE INDIRECT FOR COSTCODE = UPDATVAR ALL \
                   WITH INDIRECT;
         ELSE
             IF COND = 2 THEN     ! browse on PRINCIPL
                  BROWSE INDIRECT FOR PRINCIPL = UPDATVAR ALL \
                       WITH INDIRECT;
             ENDIF;
         ENDIF;
         BREAK;
```

```
CASE "INVENTOR":
     USE INVENTOR;
     LOAD FROM INVENTOR;      ! the inventory entry form
     IF COND = 1 THEN         ! browse on PA#
      BROWSE INVENTOR FOR PA# = UPDATVAR ALL \
                WITH INVENTOR;
     ELSE
         IF COND = 2 THEN     ! browse on SERIAL#
               BROWSE INVENTOR FOR SERIAL# = UPDATVAR ALL \
                     WITH INVENTOR;
         ENDIF;
     ENDIF;
     BREAK;

CASE "OPTAR":
     USE OPTAR;
     LOAD FROM OPTAR;              ! the OPTAR entry form
     IF COND = 1 THEN              ! browse on RECEIVED
          BROWSE OPTAR FOR RECEIVED = UPDATVAR ALL \
               WITH OPTAR;
     ELSE
          IF COND = 2 THEN         ! browse on ENTRY
               BROWSE OPTAR FOR ENTRY = UPDATVAR ALL \
                    WITH OPTAR;
          ENDIF;
     ENDIF;
     BREAK;
```

```
CASE "OTHER":
     USE OTHER;
     LOAD FROM OTHER;        ! the other funds entry form
     IF COND = 1 THEN        ! browse on COSTCODE
          BROWSE OTHER FOR COSTCODE = UPDATVAR ALL \
               WITH OTHER;
     ELSE
          IF COND = 2 THEN   ! browse on PRINCIPL
               BROWSE OTHER FOR PRINCIPL = UPDATVAR ALL \
                    WITH OTHER;
          ENDIF;
     ENDIF;
     BREAK;

CASE "PFUNDING":
     USE PFUNDING;
     LOAD FROM PFUNDING; ! the personnel funding entry form
     IF COND = 1 THEN    ! browse on LNAME
          BROWSE PFUNDING FOR LNAME = UPDATVAR ALL \
               WITH PFUNDING;
     ELSE
          IF COND = 2 THEN ! browse on LABOR#
               BROWSE PFUNDING FOR LABOR# = UPDATVAR ALL \
                    WITH PFUNDING;
          ENDIF;
     ENDIF;
     BREAK;

CASE "PHISTORY":
     USE PHISTORY;
     LOAD FROM PHISTORY; ! the personnel history entry form
     BROWSE PHISTORY FOR LNAME = UPDATVAR ALL WITH PHISTORY;
     BREAK;
```

```
  CASE "PURCHASE":
        USE PURCHASE;
        LOAD FROM PURCHASE;    ! the purchase entry form
        BROWSE PURCHASE FOR DOC# = UPDATVAR ALL WITH PURCHASE;
        BREAK;

    CASE "TVLEXP":
        USE TVLEXP;
        LOAD FROM TVLEXP;    ! the travel expenses entry form
        BROWSE TVLEXP FOR DOC# = UPDATVAR ALL WITH TVLEXP;
        BREAK;
ENDTEST;

FINISH ALL;

FORMANS = " ";

RETURN;
```

APPENDIX J
LIST OF TABLES

This appendix contains a list of the AS System's tables. Appendix C (the Data Dictionary) and the AS System's User's Manual contains additional information about the tables's fields. The inclosed information may be obtained by using KnowledgeMan's "show" command.

```
Table name    : INDIRECT
File name     : INDIRECT.ITB
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Creation Date      :  02/25/86
Modification Date :  02/25/86
Number of Records :       44

     ----------
Field : #MARK        LOGIC
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture :  (default)

     ----------
Field : ENTRY        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : STATUS       STR  25
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%25r"

     ----------
Field : PRINCIPL     STR  20
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%20r"

     ----------
Field : SPONSOR      STR  40
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%40r"

     ----------
Field : PROPOSAL     STR  130
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%130r"

     ----------
Field : PERIOD       STR  20
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%20r"

     ---------
```

```
Field : ESTIMATE     NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddddddd.dd"

     ----------
Field : DEPTIC       NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddddd.dd"

     ----------
Field : MIPR#        STR  15
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%15r"

     ----------
Field : REF#         STR  25
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%25r"

     ----------
Field : AUTH         NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddddddd.dd"

     ----------
Field : COSTCODE     STR  5
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%5r"

     ----------
Field : EXPIRE       NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : ICRECD       NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddddd.dd"

     ----------
```

```
Field : REMARKS      STR  50
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%50r"

     ----------
Field : STAFFLBR     NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddddd.dd"

     ----------
Field : ASAPPRO      NUM
Read  Access : A................
Write Access : A................
Picture : "$ddddd.dd"

     ----------
Field : DOC#         STR  15
Read  Access : A................
Write Access : A................
Picture : "%15r"

     ----------
Field : SEGMENT      STR  6
Read  Access : A................
Write Access : A................
Picture : "%6r"

     ----------
Field : SERIALS      NUM
Read  Access : A................
Write Access : A................
Picture : "dddd-dddd"

     ----------
```

```
Table name    : INVENTOR
File name     : INVENTOR.ITB
Read  Access : A...............
Write Access : A...............
Creation Date      :  12/18/85
Modification Date :  12/18/85
Number of Records :        0

     ----------
Field : #MARK        LOGIC
Read  Access : A...............
Write Access : A...............
Picture :  (default)

     ----------
Field : PA#          STR  6
Read  Access : A...............
Write Access : A...............
Picture : "%6r"

     ----------
Field : VENDOR       STR  40
Read  Access : A...............
Write Access : A...............
Picture : "%40r"

     ----------
Field : DESCRIPT     STR  50
Read  Access : A...............
Write Access : A...............
Picture : "%50r"

     ----------
Field : SERIAL#      STR  12
Read  Access : A...............
Write Access : A...............
Picture : "%12r"

     ----------
Field : PO#          STR  9
Read  Access : A...............
Write Access : A...............
Picture : "%9r"

     ----------
Field : COST         NUM
Read  Access : A...............
Write Access : A...............
Picture : "$dddddd.dd"

     ----------
```

```
Field : RECEIVED    NUM
Read  Access : A.............
Write Access : A.............
Picture : "dd-dd-dd"

     ----------
Field : ISSUE       STR  25
Read  Access : A.............
Write Access : A.............
Picture : "%25r"

     ----------
Field : LOCATION    STR  7
Read  Access : A.............
Write Access : A.............
Picture : "%7r"

     ----------
```

```
Table name    : OPTAR
File name     : OPTAR.ITB
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Creation Date      :  02/25/86
Modification Date :  02/26/86
Number of Records :        2

     ----------
Field : #MARK        LOGIC
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture :  (default)

     ----------
Field : ENTRY        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : RECEIVED     NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : AUTH         NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$dddddd.dd"

     ----------
Field : TAUTH        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$dddddd.dd"

     ----------
Field : ACTUAL       NUM
Read  Access : A...............
Write Access : A...............
Picture : "$dddddd.dd"

     ----------
Field : TACTUAL      NUM
Read  Access : A...............
Write Access : A...............
Picture : "$dddddd.dd"

     ----------
```

```
Field : DIFFER
Virtual field : ((ACTUAL - AUTH))
Read  Access : A..............
Picture :  (default)

     ----------
Field : TDIFFER
Virtual field : ((TACTUAL - TAUTH))
Read  Access : A..............
Picture :  (default)

     ----------
```

```
Table name    : OTHER
File name     : OTHER.ITB
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Creation Date      :  02/25/86
Modification Date :  02/25/86
Number of Records :        6

     ----------
Field : #MARK        LOGIC
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture :  (default)

     ----------
Field : ENTRY        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : STATUS       STR  25
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%25r"

     ----------
Field : EXPIRE       NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : PRINCIPL     STR  20
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%20r"

     ----------
Field : SPONSOR      STR  35
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%35r"

     ----------
Field : PROPOSAL     STR  130
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%130r"

     ----------
```

```
Field : COSTCODE     STR  5
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%5r"

     ----------
Field : AUTH         NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$dddddd.dd"

     ----------
Field : REMARKS      STR  50
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%50r"

     ----------
```

```
Table name    : PFUNDING
File name     : PFUNDING.ITB
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Creation Date      :  02/27/86
Modification Date :  02/27/86
Number of Records :       314

     ----------
Field : #MARK        LOGIC
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture :  (default)

     ----------
Field : ENTRY        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : LNAME        STR  25
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%25r"

     ----------
Field : FNAME        STR  15
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%15r"

     ----------
Field : FUNDFROM     NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : FUNDTO       NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : DAYS         STR  5
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%5r"

     ----------
```

```
Field : COSTCODE     STR  5
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%5r"

     ----------
Field : AMOUNT       NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddddd.dd"

     ----------
Field : LABOR#       NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : CODE         STR  3
Read  Access : A...............
Write Access : A...............
Picture : "%3r"

     ----------
Field : HOURS        NUM
Read  Access : A...............
Write Access : A...............
Picture : "d"

     ----------
Field : RATE         NUM
Read  Access : A...............
Write Access : A...............
Picture : "$ddd.dd"

     ----------
Field : HRSWK        NUM
Read  Access : A...............
Write Access : A...............
Picture : "dd"

     ----------
```

```
Table name    : PHISTORY
File name     : PHISTORY.ITB
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Creation Date      :  02/27/86
Modification Date :  02/27/86
Number of Records :        27

     ----------
Field : #MARK        LOGIC
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture :  (default)

     ----------
Field : ENTRY        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : LNAME        STR  25
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%25r"

     ----------
Field : FNAME        STR  15
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%15r"

     ----------
Field : DOB          NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : EFFECTIV     NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : STATUS       STR  6
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%6r"

     ----------
```

```
Field : BILLET#      STR  4
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%4r"

     ----------
Field : JOBTITLE     STR  30
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%30r"

     ----------
Field : ES           NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : WORKUNIT     STR  10
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%10r"

     ----------
Field : SUPERVSR     STR  25
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%25r"

     ----------
Field : PD#          STR  5
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%5r"

     ----------
Field : SERIES       NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "ddddd"

     ----------
Field : GRADE        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd"

     ----------
```

```
Field : STEP          NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd"

     ----------
Field : SALARY        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddddd.dd"

     ----------
Field : HRSWK         NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd"

     ----------
Field : EXPIRE        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : REMARKS       STR  100
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%100r"

     ----------
Field : RATE
Virtual field : (((SALARY / 2087) * 8 * 1.30000)))
Read  Access : ABCDEFGHIJKLMNOP
Picture : "$ddd.dd"

     ----------
```

```
Table name    : PURCHASE
File name     : PURCHASE.ITB
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Creation Date      :  02/11/86
Modification Date :  02/28/86
Number of Records :       268

     ----------
Field : #MARK        LOGIC
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture :  (default)

     ----------
Field : ENTRY        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : DOC#         STR  10
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%10r"

     ----------
Field : COSTCODE     STR  5
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%5r"

     ----------
Field : VENDOR       STR  40
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%40r"

     ----------
Field : DESCRIPT     STR  50
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%50r"

     ----------
Field : STOCK#       STR  20
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%20r"

     ----------
```

```
Field : ESTIMATE     NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddddd.dd"

     ----------
Field : PRIORITY     NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd"

     ----------
Field : RDD          NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : CODE         STR  3
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%3r"

     ----------
Field : REQUESTR     STR  25
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%25r"

     ----------
Field : PO#          STR  9
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%9r"

     ----------
Field : RECEIVED     NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : FIRM         NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddddd.dd"

     ----------
```

```
Field : SERIAL#      STR  12
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%12r"

     ----------
Field : PA#          STR  17
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%17r"

     ----------
Field : ISSUE        STR  25
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%25r"

     ----------
Field : LOCATION     STR  7
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%7r"

     ----------
Field : REMARKS      STR  50
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%50r"

     ----------
Field : QTY          STR  3
Read  Access : A...............
Write Access : A...............
Picture : "rrr"

     ----------
```

```
Table name    : TVLEXP
File name     : TVLEXP.ITB
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Creation Date      :  02/28/86
Modification Date :  02/28/86
Number of Records :       133

     ----------
Field : #MARK        LOGIC
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture :  (default)

     ----------
Field : DATEDEP      NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : REQUESTR     STR  20
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%20r"

     ----------
Field : LOCATION     STR  25
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%25r"

     ----------
Field : CODE         STR  3
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%3r"

     ----------
Field : ESTIMATE     NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$dddd.dd"

     ----------
Field : COSTCODE     STR  5
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%5r"

     ----------
```

```
Field : DOC#          STR  10
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "%10r"

     ----------
Field : ADVANCE       NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddd.dd"

     ----------
Field : DATERET       NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : CLAIM         NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : FIRM          NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "$ddddd.dd"

     ----------
```

```
Field : ENTRY        NUM
Read  Access : ABCDEFGHIJKLMNOP
Write Access : ABCDEFGHIJKLMNOP
Picture : "dd-dd-dd"

     ----------
Field : DOV#         STR  5
Read  Access : A...............
Write Access : A...............
Picture : "rrrrr"

     ----------
```

VII. LIST OF REFERENCES

1. Boehm, Barry W., Software Engineering Economics, pp. 37, Prentice Hall, 1981.

2. Henderson, John and Ingraham, Robert, Prototyping for DSS: A Critical Appraisal, working paper, pp. 1, Department of Management, Florida State University, 1979.

3. Keen, Peter G.W. and Morton, Michael Scott, Decision Support Systems: An Organizational Perspective, pp. 214, Addison-Wesley Publishing Co., 1978.

4. Bennett, John L., edt, Building Decision Support Systems, pp. 151, Addison-Wesley, 1982.

5. Naumann, Justus D. and Jenkins, A. Milton, "Prototyping: The New Paradigm for Systems Development," MIS Quarterly, pp. 29-43, September 1982.

6. Renner, Richard B., Information Requirements Analysis: An Application, Master's Thesis, Naval Postgraduate School, Monterey, California, March 1984.

7. De Marco, T., Structure Analysis and System Specification, pp. 3, Prentice-Hall, 1979.

8. Ginzberg, Michael, A Process Approach to Management Science Implementation, Ph.D. Dissertation, pp. 1, M.I.T., 1975.

9. Earl, M.J., "Prototype Systems for Accounting, Information and Control," Accounting, Organizations and Society, pp. 3, March 1978.

10. Henderson, pp. 2.

VIII. BIBLIOGRAPHY

Alavi, Maryam, User-Developed DSS: Steps Toward Quality Control, paper presented at the NYU Symposium on Integrating Systems for End Users, New York, New York, 22-24 May 1985.

Alavi, Maryam and Henderson, John C., "An Evolutionary Strategy for Implementing a Decision Support System," Management Science, v. 27, n. 11, pp. 1309-1323, November 1981.

Alavi, Maryam and Napier, Albert, "An Experiment in Applying the Adaptive Design Approach to DSS Development," Information and Management, v. 7, n. 1, pp. 21-28, February 1984.

Alter, Steven, "A Taxonomy of Decision Support Systems", Sloan Management Review, v. 19, n. 1, pp. 39-56, Fall 1977.

Alter, Steven, Decision Support Systems: Current Practice and Continuing Changes, Reading, Massachusetts, Addison-Wesley, 1980.

De, Prabuddha and Haseman, William D., "An Integrated Database Design for Accounting Systems," Information Processing & Management, v. 20, n. 4, pp. 507-518, 1984.

Denise, Richard M., "Technology for the Executive Thinker," Datamation, v. 29, n. 6, pp. 207-211.

Huber, G.P., "The Nature of Organizational Decision Making and the Design of Decision Support Systems," MIS Quarterly, v. 5, n. 2, pp. 1-10, June 1981.

Keen, Peter G.W., "Computer-Based Decision Aids: The Evaluation Problem," Sloan Management Review, v.16, no. 3, pp. 17-29, Spring 1975.

Keen, Peter G.W., "DSS: An Executive Mind-Support System," Datamation, v. 25, n. 12, pp. 117-122, November 1979.

Keen, Peter G.W., "Decision Support Systems: Translating Analytic Techniques into Useful Tools," Sloan Management Review, v. 21, n. 3, pp. 33-44, Spring 1980.

Keen, Peter G.W., "Value Analysis: Justifying Decision Support Systems," MIS Quarterly, v. 5, n. 1, pp. 1-15, March 1981.

Kingston, Paul L., "Generic Decision Support Systems," Managerial Planning, v. 29, n. 5, pp. 7-11, March/April 1981.

Lasden, Martin, edt., "Computer-Aided Decision-Making," Computer Decisions, pp. 156-172, November 1982.

Lasden, Martin, edt., "Enriching the Decision-Making Process," Computer Decisions, pp. 244-258, November 1983.

Mann, R.I., "A Contingency Model for User Involvement in DSS Development," MIS Quarterly, v. 8, n. 1, pp. 27-38, pp. 27-38, March 1984.

McCosh, A.M., Rahman, M. and Earl, M.J., Developing Managerial Information Systems, Halsted Press, 1981.

Patel, Has, "How to Upgrade a DBMS," Datamation, v. 27, n. 10, pp. 127-132, September 1981.

Radhakrishan, T., Grossner, C. and Benoliel, M., "Design of an Interactive Data Retrieval System for Casual Users," Information Processing & Management, v.18, n. 1, pp. 23-32, 1982.

Rogers, G. R., "A Simple Architecture for Consistent Application Program Design," IBM Systems Journal, v. 22, n. 3, pp. 199-213, 1983.

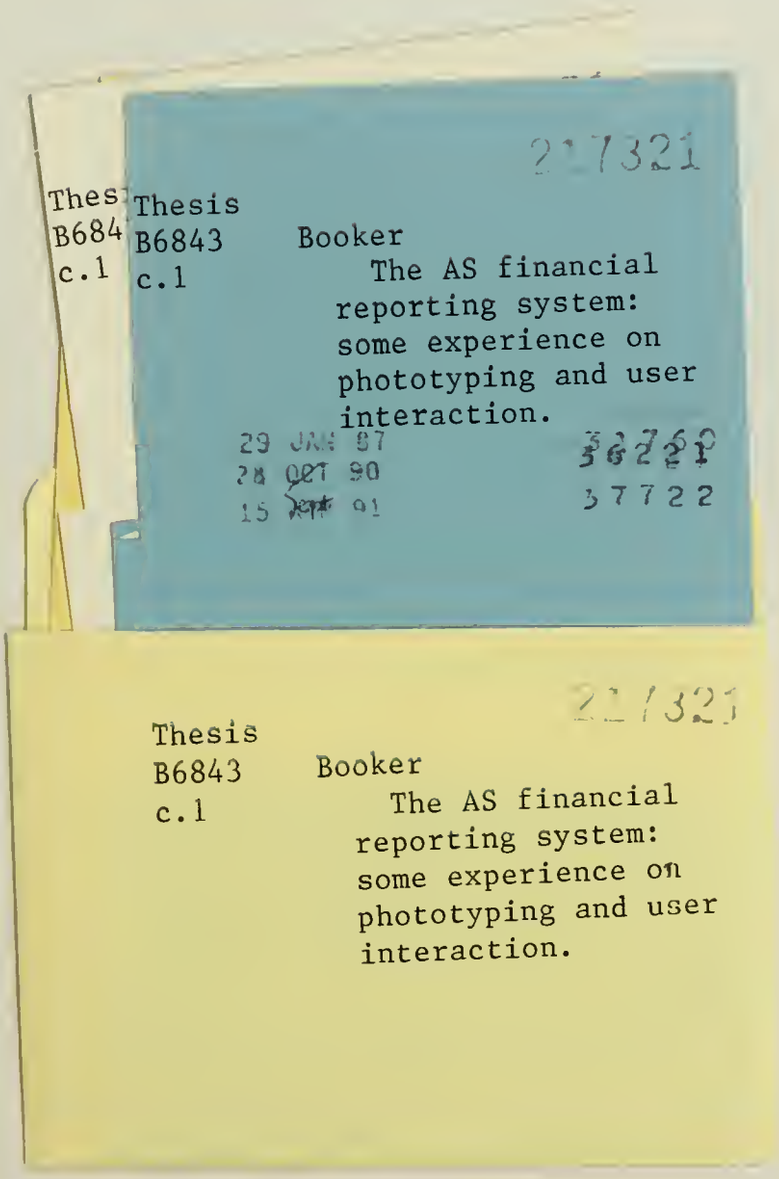
Spraque, R.H., "A Framework for the Development of Decision Support Systems," MIS Quarterly, v. 4, n. 4, December 1979.

Vierck, R.K., "Decision Support Systems: An MIS Manager's Perspective," MIS Quarterly, v. 5, n. 4, pp. 35-48, December 1981.

Welsch, Gemma M., "Successful Implementation of Decision Support Systems: The Role of the Information Transfer Specialist," American Inst. for Decision Sciences, pp. 206-208, November 1981.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3.	Computer Technology Programs, Code 37 Naval Postgraduate School Monterey, California 93943-5000	2
4.	Department Chairman, Code 54 Naval Postgraduate School Monterey, California 93943-5004	1
5.	Beth Buttles-Miller, Code 54 Naval Postgraduate School Monterey, California 93943-5004	2
6.	Assistant Professor Tung Bui, Code 54BD Naval Postgraduate School Monterey, California 93943-5004	1
7.	Professor Norman F. Schneidewind, Code 54SS Naval Postgraduate School Monterey, California 93943-5004	2
8.	CAPT Ronald L. Booker, USMC 4115 Carozza Court Temple Hills, Maryland 20748	1


Thesis
B6843
c.1
217321
Thesis
B6843
c.1
Booker
The AS financial reporting system: some experience on phototyping and user interaction.
29 JAN 87
28 OCT 90
15 SEP 91
36760
36221
37722
217321
Thesis
B6843
c.1
Booker
The AS financial reporting system: some experience on phototyping and user interaction.